Résolution de problèmes réalistes de tournées à flotte hétérogène en milieu urbain : vers un solveur adaptatif mêlant recherche opérationnelle et apprentissage automatique

Flavien Lucas

DOCTORAT BRETAGNE LOIRE MATHSTIC

Université Bretagne Sud

THÈSE DE DOCTORAT DE

L'UNIVERSITE BRETAGNE SUD

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par
Flavien LUCAS

Résolution de problèmes réalistes de tournées à flotte hétérogène en milieu urbain :

vers un solveur adaptatif mêlant recherche opérationnelle et apprentissage automatique.

Thèse présentée et soutenue à Lorient, le 19 Novembre 2020
Unité de recherche : lab-STICC, UMR 6285
Thèse N° : 569

Rapporteurs avant soutenance :

Latifa Oukhellou	Directrice de recherche, IFSTTAR
Daniele Vigo	Professeur, Université de Bologne

Composition du Jury :

Président :	Fabien LEHUÉDÉ	Professeur, IMT Atlantique
Examinateurs :	Latifa OUKHELLOU	Directrice de Recherche, Université Gustave Eiffel
	Daniele VIGO	Professeur, Université de Bologne
	Christine SOLNON	Professeure, INSA Lyon
	Christian PRINS	Professeur, Université de Technologie de Troyes
Dir. de thèse :	Marc SEVAUX	Professeur, Université de Bretagne Sud
Co-dir. de thèse :	Romain BILLOT	Professeur, IMT Atlantique

Remerciements

Je tiens à remercier les rapporteurs et membres du jury, Fabien Lehuédé, Latifa Oukhellou, Daniele Vigo, Christine Solnon et Christian Prins, pour leur présence pendant la soutenance, leur lecture du manuscrit, ainsi que pour la qualité des échanges scientifiques qui en ont découlé.

Je remercie Marc Sevaux et Romain Billot pour leur confiance, leur soutient et conseils tout au long de ces trois ans, malgré un emploi du temps très chargé.

Je remercie mes anciens collègues de bureau, dont Abdessamad, Yohann, Paul et Julien pour leurs conseils, ainsi que Quentin, dont je souhaite une bonne continuation pour la thèse. Je remercie tous les membres du Lab-STICC à Lorient pour les discussions liées à d'innombrables sujets.

Je remercie également les proches sur Lorient, Dihya, Alix, Clémantyne, Alex, Erwan, Florent, Cynthia, ainsi que tous les autres. Je remercie Malik pour les balades en vélo, et Maxime, qui m'a permis d'entrer dans cet univers qu'est la thèse.

Je remercie Virginie et Florence pour leur aide et soutient au cours de ces trois ans.

Je remercie les collègues de l'équipe DECIDE, et particulièrement les brestois (Patrick, Philippe, Mehrdad, Bastien, Atena, Antoine, Arwa et de nombreux autres) pour leur accueil lors de mes visites à IMT Atlantique, ainsi que lors de séminaires d'équipe.

Je remercie Kenneth Sörensen pour son accueil à l'université d'Anvers, ainsi que les collègues rencontrés sur place, (Trijntje, Tyché, Kevin, Mauricio, Luana, Ying, David, Michele, Lissa, Babiche, Floor, Stiene ...).

Je remercie Gwenaël, Pierre et Philippe pour leur collaboration tout au long des ces trois ans.

Je tiens à adresser des remerciements particuliers à Alexis et Victor pour les innombrables moment de soutient, ainsi que pour les nombreuses discussions, passées et à venir.

Je tiens enfin à remercier ma famille qui a toujours cru en moi.

TABLE DES MATIÈRES

INTRODUCTION

Contexte

Avec l'avènement du commerce électronique, la demande de livraison à domicile ou en point-relais a considérablement augmenté ces dernières années tandis que que de plus en plus de contraintes pèsent sur les livreurs dans l'organisation de leur tournée : congestion des centres urbains, zones de circulation restreinte etc. Dans le domaine de la recherche opérationnelle, la résolution des problèmes de tournées constitue un défi emblématique étudié dès les années 1960 et considérablement enrichi depuis les années 2000. Initialement développées sur des problèmes simples (nombre de clients limité, distances euclidiennes), de nombreuses variantes ont permis d'améliorer le réalisme des instances étudiées, en particulier en milieu urbain où la congestion n'est pas homogène, et dépendante du véhicule utilisé. Cependant, encore peu de méthodes sont compatibles avec l'ensemble des nombreuses variantes existantes, et celles-ci perdent en efficacité (qualité de la solution moindre, et/ou temps de calculs long) avec l'augmentation du nombre de clients.

Ces dernières décennies, la quantité de données publiques disponible a considérablement augmenté. Cet apport massif d'informations nouvelles a permis le retour sur le devant de la scène scientifique des méthodes d'apprentissage automatique, ou machine learning. L'exemple le plus connu est la victoire au jeu de go d'un algorithme fondé sur les réseaux de neurones profonds contre un joueur professionnel. Ce nouvel intérêt, aussi bien de la communauté scientifique que de la société en général, a abouti au développement de collaborations inter-disciplinaires depuis différentes disciplines vers l'apprentissage automatique. Néanmoins, à l'image du *buzz* big data il y a quelques années, il s'avère nécessaire de comprendre et d'évaluer la pertinence de telles approches comparées à un état de l'art plus ancien et aux réels besoins du domaine de l'optimisation des tournées.

C'est dans ce contexte qu'est apparue l'idée d'utiliser des méthodes d'analyse statistique ou d'apprentissage pour améliorer l'efficacité des métaheuristiques. Malgré de nombreuses recherches sur le domaine du transport, nous ne connaissons toujours pas clairement ce qui définit une bonne solution, notamment pour les problèmes complexes issus de données réelles. En effet, en exploitant les historiques de données, il semble pos-

sible de mieux caractériser les solutions afin d'améliorer la résolution de problèmes de tournées, en réduisant le temps d'exécution des méthodes ou en améliorant la qualité des solutions obtenues.

Problématique de la thèse

Le contexte au commencement de la thèse a donc mené aux questions suivantes :

— Question de recherche 1 : comment disposer d'un solveur adaptatif et capable de résoudre de très grandes instances ?

— Question de recherche 2 : comment caractériser une solution d'un problème de tournée ?

— Question de recherche 3 : comment guider la recherche d'une bonne solution grâce à l'apprentissage automatique ?

Description du manuscrit

Dans ce présent manuscrit, le lecteur a accès à une brève explication du problème de tournée de véhicules, ainsi que ses variantes. Un état de l'art, limité à certaines variantes et méthodes sera ensuite décrit afin de situer les avancées de la recherche sur ces problèmes. Enfin, cette partie se terminera par un résumé des performances de chacune des méthodes.

Cette première section est ensuite suivie par une description générale des enjeux de l'apprentissage statistique et des méthodes associées, avant d'être complétée par un état de l'art des méthodes hybrides, combinant recherche opérationnelle et apprentissage automatique.

La section 3 succédant à l'état de l'art concerne la description d'un nouveau solveur, développé pendant les trois ans de thèse, ainsi que ses performances sur des jeux d'instances connus pour leur difficulté. Celui-ci n'utilise pas d'apprentissage automatique et sert de base pour la suite des travaux.

La section 4 "*Caractérisation descriptive d'une solution*" se concentre sur la partie analyse statistique. Après une étude graphique des solutions du VRP, nous nous intéresserons à des méthodes automatisables permettant de déduire des règles. Enfin, nous terminerons par une méthode permettant de comprendre les caractéristiques nécessaires

au développement de règles sur le fonctionnement d'une bonne solution sur un groupe d'instance donné.

La section 5 "*Méthode hybride fondée sur l'extraction de règles*" explique en détail le fonctionnement d'un nouveau solveur hybride, utilisant les informations obtenues dans la section précédente pour améliorer notre solveur. Après une brève description de l'idée de départ, et une explication détaillée de ce nouveau solveur, cette section se poursuivra par un détail des expérimentations effectuées, avant de finir sur les performances de celui-ci.

Enfin, ce document se conclura par un résumé des contributions de la thèse et des pistes émergeant à l'issue de celle-ci.

La figure 1 résume l'organisation du manuscrit.

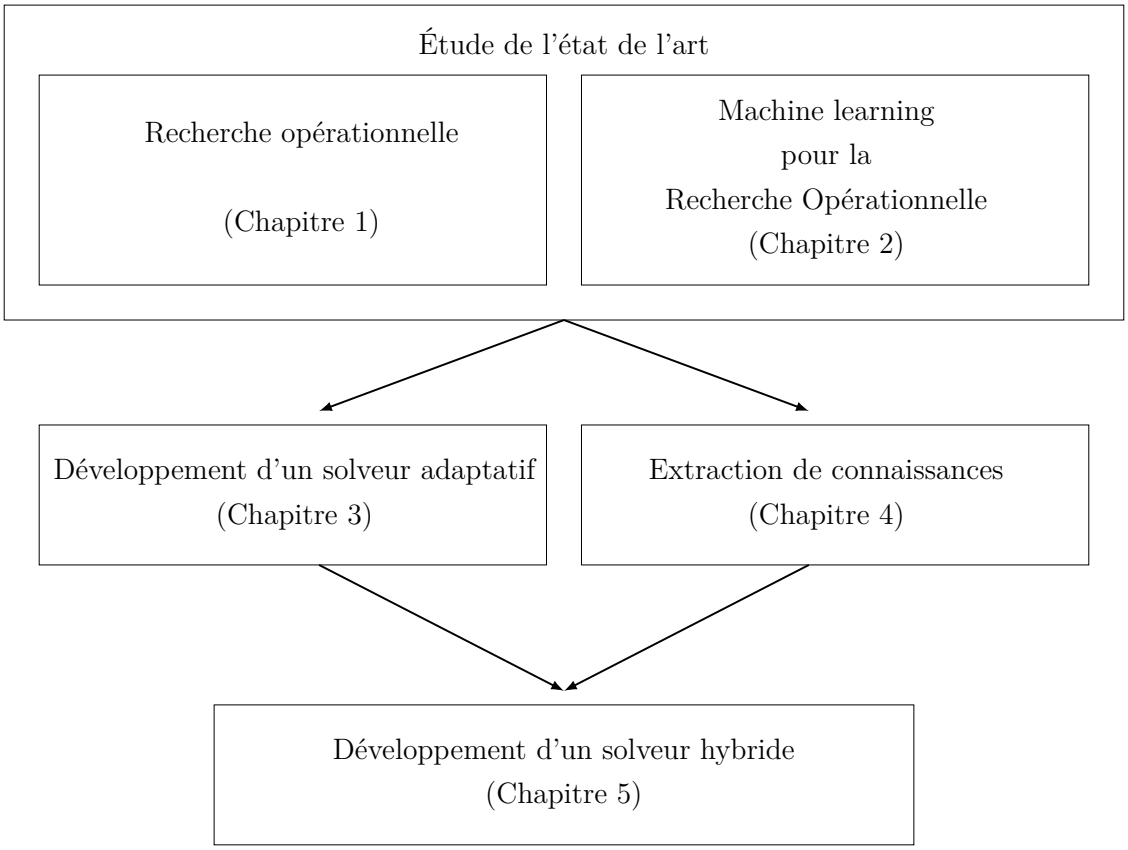


FIGURE 1 – Organisation du manuscrit

Chapitre 1

Résolution des problèmes de VRP

Ce chapitre explique en détail les différents problèmes de livraison, du plus simple, au plus complexe. Nous commencerons par résumer l'évolution dans la résolution du premier problème de transport traité dans la littérature. Ensuite, nous expliquerons le problème de transport le plus étudié, ainsi que la liste de ses très nombreuses variantes. Enfin, nous ferons un inventaire des méthodes utilisées pour traiter ces variantes, complété par un bilan des performances de ces méthodes sur différents jeux d'instances.

Les méthodes utilisées dans les différents solveurs développés seront expliquées plus en détails, dans les chapitres correspondants.

Sommaire

1.1 Description des problèmes de tournées

1.1.1 Le problème du voyageur de commerce (TSP)

L'exemple le plus ancien et le plus connu est le problème du voyageur de commerce (*Traveling Salesman Problem* - TSP). Celui-ci peut être formulé de la manière suivante : "*Soit une liste de villes dont la distance deux à deux est connue, quel est le plus court chemin permettant de visiter une et une seule fois chaque ville, et revenir au point de départ ?*". Ce problème est très connue en théorie des graphes et consiste à trouver le plus court cycle hamiltonien.

FIGURE 1.1 – Exemple d'instance du TSP
(© les contributeurs OpenStreetMap)

Ce problème est $\mathcal{NP}$-difficile, ce qui signifie qu'il n'existe pas, à ce jour, d'algorithme permettant de résoudre en temps polynomial toute instance de ce problème, et il n'est pas certain qu'un tel algorithme existe.

Historique

Ce problème a été étudié pour la première fois par Sir W.R. Hamilton et T. Kirkman dans les années 1800. Le but était alors de réussir à relier tous les sommets d'un icosaèdre. L'application au domaine du transport n'a été portée qu'à partir de 1930, par H. Whitney et M. Flood. Ce n'est que dans les années 50 que le modèle linéaire a été imaginé par

Dantzig *et al.* (1954). Ils ont également développé une première méthode de coupe pour faciliter la résolution du problème. Cette méthode a été testée sur un problème à 49 villes, qui a nécessité 26 coupes.

Deux décennies plus tard, Grötschel (1980) a résolu une instance de 120 clients, sur une carte de l'Allemagne de l'ouest. En 1987, une instance de 2392 clients a été résolue par Padberg et Rinaldi (1990).

Applegate, Bixby, et Chvatal ont résolu en exacte dans les années 90 et 2000 des instances de tailles de plus en plus importantes : 7397 en 1994 (Applegate et Bixby), 13509 en 1998 (Applegate *et al.*), 15 112 en 2001 (Applegate *et al.*), 24 978 en 2004, et finalement, 85900 en 2006 (Applegate *et al.*).

En 2013, une instance de presque deux millions de villes a été traitée par une méthode approchée (Helsgaun). La valeur de la solution obtenue a un coût de moins de 0.05 % supérieur à celui de l'optimum.

Avec les progrès techniques (matériel informatique toujours plus performant), et méthodologique (les algorithmes réduisent au fil des années considérablement le nombre de solutions étudiées), des instances de taille de plus en plus grande ont pu être résolues au fil du temps. La figure 1.2, dont l'axe des ordonnées est en échelle logarithmique, montre cette avancée exponentielle.

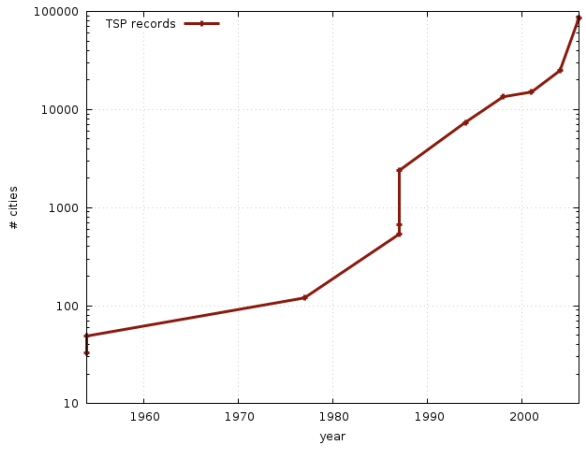


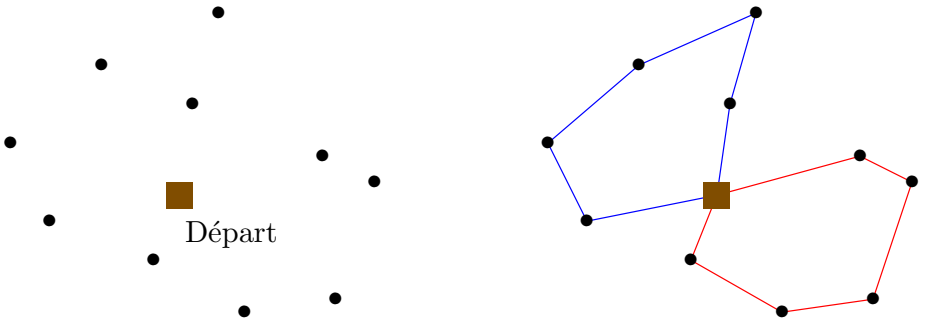
FIGURE 1.2 – Evolution de la taille des problèmes traités

Afin de faciliter les expérimentations, la plupart de ces instances sont au format TSPlib dont on peut trouver la description dans la publication de Reinelt (1991). Le lecteur intéressé pourra se référer à la publication de Applegate *et al.* (2006) qui recense la plupart des méthodes connues.

1.1.2 Le problème de tournées de véhicules (CVRP)

Étudié pour la première fois par Dantzig et Ramser (1959), le problème de tournées de véhicules (Vehicle Routing Problem - VRP) est le suivant : *Soit un ensemble de véhicules dans un dépôt, et soit un ensemble de clients, quel véhicule doit livrer quel client, et dans quel ordre, de façon à livrer chaque client en un minimum de déplacement ?* Ce problème est donc composé de deux sous-problèmes :

1. Quelle est la meilleure affectation clients-véhicules ? Trouver la réponse à ce problème est équivalent à résoudre un problème de bin-packing.
2. Quel est le plus court trajet à effectuer pour chaque véhicule afin de livrer l'ensemble des clients ? Résoudre ce problème est équivalent à résoudre un problème de TSP.



(a) Situation initiale (b) Exemple de solution

FIGURE 1.3 – Exemple de VRP

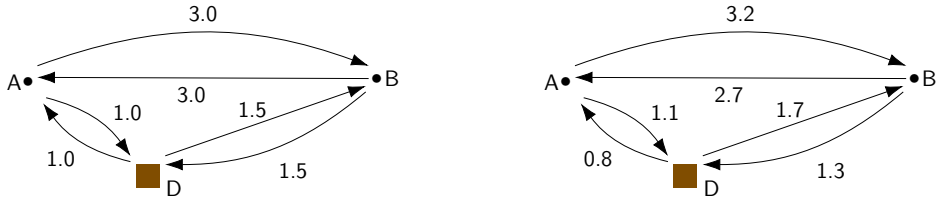
Il est commun d'attribuer une limite de capacité de transport à chaque véhicule, sinon, le problème est celui du TSP avec dépôt. On parle alors de *Capacitated Vehicle Routing Problem* (CVRP).

1.1.3 Les variantes du problème de tournées

Les problèmes de tournées ont fait l'objet de milliers de publications depuis l'article de Dantzig et Ramser (1959). Ainsi, de nombreuses variantes ont été créées. Afin de mieux comprendre l'ensemble des possibilités d'étude, les prochains paragraphes vont présenter plusieurs de ces variantes.

Les problèmes de tournées asymétriques Ici, la matrice de distance n'est pas symétrique. Ainsi, la distance pour aller du client i au client j n'est pas forcément identique

à celle pour aller du client j au client i. La présence de symétries permet de faciliter la résolution, que ce soit via des méthodes exactes, ou approchées. Lors de la résolution d'instances asymétriques, non seulement, l'affectation client-véhicules, et l'ordre de livraison sont importants, mais désormais, le sens de livraison compte également. Par exemple, dans la figure 1.4(b), le trajet $D \to A \to B \to D$ a un coût de 5.3, alors que le trajet $D \to B \to A \to D$ a un coût de 5.5, alors que dans le cas symétrique (figure 1.4(a)), le coût est de 5.5 dans les deux sens.



(a) Graphe symétrique (b) Graphe asymétrique

FIGURE 1.4 – Graphes symétriques et asymétriques

Les problèmes de tournées avec distances non euclidiennes Dans la plupart des instances de la littérature, les clients et le dépôt sont représentés uniquement par des points. La distance pour aller d'un point à l'autre correspond à la distance euclidienne. Cette distance a plusieurs propriétés, dont la symétrie (voir paragraphe précédent) et l'inégalité triangulaire : soit i, j, k trois clients, et $d(i, j)$ la distance pour aller du client i au client j, on a : $d(i, j) \leq d(i, k) + d(k, j)$.

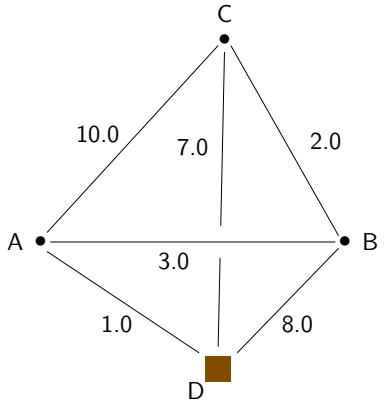


FIGURE 1.5 – Exemple de graphe non euclidien

Cependant, en pratique cette inégalité n'est pas toujours respectée. Par exemple, dans la figure 1.5, le trajet $D \to A \to B \to C$ est plus court que le trajet $D \to C$.

Certains solveurs sont basés, au moins en partie, sur l'inégalité triangulaire qui fournit en effet des propriétés intéressantes, dont l'absence de croisement au sein d'une tournée. Les solveurs sont donc pensés pour utiliser ces propriétés. Ainsi, les méthodes à base de recherche locale utilisent très fréquemment le mouvement 2-opt, réputé pour éliminer les croisements propres à chaque tournée. L'absence d'inégalité triangulaire réduit l'efficacité de ces méthodes, et les optima locaux obtenus sont parfois visuellement peu intuitifs. L'influence de cette particularité est étudiée plus en détail dans le chapitre 4.

Les flottes hétérogènes Il existe plusieurs moyens de livraison : deux roues motorisés, non motorisés, automobiles, poids lourds, etc. Chaque véhicule possède ses propres forces et faiblesses et leur utilisation doit donc être réfléchie. Les problèmes avec flottes hétérogènes prennent en compte le cas où les véhicules disponibles ne sont pas identiques. La différence peut se faire sur la vitesse de circulation, la capacité du véhicule, le coût d'utilisation, ou les clients accessibles.

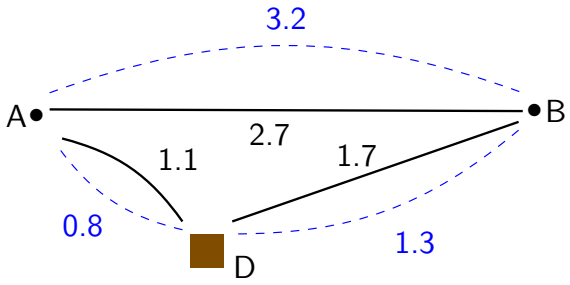


FIGURE 1.6 – Exemple de graphe hétérogène

La figure 1.6 montre un exemple de graphe avec deux catégories de véhicules. Les arêtes en noir (trait plein) correspondent aux distances entre les nœuds pour le premier type de véhicule, celles en bleus (pointillé) pour le deuxième type de véhicule.

Dans cette variante, il convient donc de bien choisir quelle catégorie de véhicules livre chaque client, accentuant encore plus l'importance de l'affectation clients/véhicules.

Livraisons avec fenêtres de temps Il est fréquent que les clients ne soient disponibles que sur un intervalle de temps spécifique. Il est donc nécessaire de les livrer pendant cet intervalle. Un véhicule arrivant à destination avant l'intervalle de disponibilité du client devra attendre avant de le livrer.

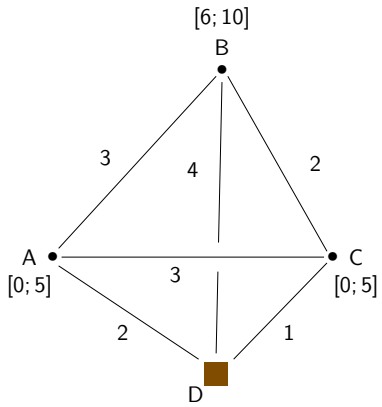


FIGURE 1.7 – Exemple de problème avec fenêtres de temps

Exemple : Soit la tournée $D \to A \to B \to C \to D$ avec les distances et fenêtres de temps indiquées dans la figure 1.7. Le véhicule part de D au temps $t = 0$ puis arrive en A à $t = 2$. Cette valeur étant comprise dans la fenêtre de temps du client A, celui-ci est livré immédiatement et repart aussitôt. Il arrive en B à $t = 5$. Le client B n'étant disponible qu'à partir du temps $t = 6$, le véhicule attend une unité de temps avant de le livrer, puis repart. Il arrive en C au temps $t = 6+2 = 8$. Le véhicule arrive donc en C avec 3 unités de temps de retard. La solution est donc infaisable. Dans le cas de fenêtre de temps souples (où *soft time windows*), la solution est toujours réalisable, mais subit des pénalités à cause du retard.

Ces contraintes temporelles font qu'il est parfois impossible de livrer des clients proches géographiquement, car leurs intervalles de disponibilités ne sont pas compatibles. Une bonne solution lorsque ces contraintes sont relaxées peut donc entraîner une mauvaise solution sous ces contraintes, voire mener à une solution infaisable.

Les problèmes dépendant du temps Le trafic routier est extrêmement variable au cours d'une journée. Ainsi, certains axes routiers sont congestionnés en début et fin de journée, mais pas en milieu de journée. D'autres zones géographiques, correspondant majoritairement aux routes reliant les zones habitées et les zones industrielles sont congestionnées dans une direction en début de journée, et dans l'autre direction le soir.

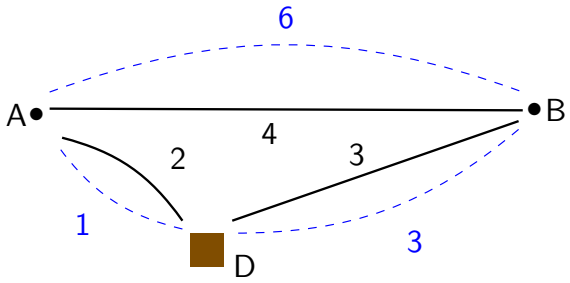


FIGURE 1.8 – Exemple de graphe dépendant du temps

Exemple : soit le graphe avec deux fonctions de coût selon l'heure de départ, présenté par la figure 1.8. Dans cet exemple, la distance entre deux sommets est représentée par le graphe en noir (trait plein) si le temps de départ est inférieur ou égal à 4, et représentée par le graphe en bleu (pointillé) si le temps de départ est supérieur à 4. Ainsi, la tournée $D \to A \to B \to D$ a un coût de 9. En effet, le véhicule part de D au temps $t = 0$, le coût de l'arête $D \to A$ est donc de 2. Comme $t = 2$, le coût de l'arête $A \to B$ est de 4. Désormais, $t = 6$, le coût de l'arête $B \to D$ est donc de 3.

Nous pouvons également remarquer que la tournée $D \to B \to A \to D$ a un coût de 10. Les problèmes dépendant du temps sont donc asymétriques.

Autres variantes Les problèmes de tournées peuvent donc avoir de nombreuses variantes, dont seule une partie a été présentée ici. Ces variantes pouvant se cumuler (et ainsi avoir des problèmes non euclidiens, avec fenêtre de temps, et des flottes hétérogènes), il y a donc une explosion combinatoire de variantes possibles.

1.2 Résolution du CVRP

1.2.1 Méthodes exactes

Cette session présente les méthodes avec preuve d'optimalité les plus connues pour résoudre des problèmes du type VRP.

Le premier réflexe en optimisation combinatoire, est de vérifier s'il est possible de résoudre le problème en modélisant son problème linéaire associé, pour ensuite le résoudre avec la méthode du simplexe (Nash, 2000). Un des modèles linéaires les plus connus pour résoudre VRP a été modélisé par Fisher et Jaikumar (1981), car elle montre bien la combinaison des contraintes de Bin Packing et de TSP.

Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe complet et non dirigé. Notons $(v_0, \ldots, v_n)$ l'ensemble des sommes du graphe, où v_0 représente le dépôt et $V \backslash v_0$ représente l'ensemble des clients à livrer. Les arêtes $(i, j) \in \mathcal{E}$ représentent les déplacements possibles entre les nœuds i et j. Notons c_{ij} le coût pour aller du nœud i au nœud j et Q la capacité des véhicules et q_i la demande du client i. Considérons les variables suivantes :

— x_{ijk} est une variable binaire indiquant si le véhicule k a parcouru l'arête (i, j)

— y_{ik} indique si le client i a été livré par le véhicule k

Le modèle linéaire développé par Fisher et Jaikumar (1981) est le suivant :

$$\text{Min.} \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=1}^{m} c_{ij} x_{ijk} \tag{1.1}$$

$$\text{s.c} \sum_{k=1}^{m} y_{ik} = 1 \qquad i = 1, \ldots, n \tag{1.2}$$

$$\sum_{k=1}^{m} y_{0k} \leq m \tag{1.3}$$

$$\sum_{i=1}^{n} q_i y_{ik} \leq Q \qquad k = 1, \ldots, m \tag{1.4}$$

$$\sum_{j=0}^{n} x_{ijk} = \sum_{j=0}^{n} x_{jik} = y_{ik} \qquad i = 0, \ldots, n;\ k = 1, \ldots, m \tag{1.5}$$

$$\sum_{v_i \in S} \sum_{v_j \in S} x_{ijk} \leq |S| - 1 \qquad k = 1, \ldots, m;\ S \subseteq V \backslash 0;\ |S| \geq 2 \tag{1.6}$$

$$y_{ik} \in \{0, 1\} \qquad i = 0, \ldots, n;\ k = 1, \ldots, m \tag{1.7}$$

$$x_{ijk} \in \{0, 1\} \qquad i = 0, \ldots, n;\ j = 0, \ldots, n;\ k = 1, \ldots, m \tag{1.8}$$

FIGURE 1.9 – modèle linéaire le plus utilisé pour présenter le problème du CVRP

L'objectif (1.1) est la minimisation du coût des arêtes utilisées. La partie Bin Packing est représentée par les contraintes (1.2) à (1.4) tandis que les contraintes (1.5) et (1.6) représentent les contraintes de flot et d'élimination de sous-tours, associées au TSP. Enfin, les contraintes (1.7) et (1.8) s'assurent du respect du domaine des variables.

L'algorithme du simplexe donnant une solution linéaire, la solution retournée n'est en général pas réalisable. Il est donc nécessaire de combiner cet algorithme avec une autre méthode de résolution pour obtenir une solution réalisable.

La génération de colonnes

Il existe de nombreux problèmes de recherche opérationnelle dont la modélisation sous forme de programme linéaire contient un nombre exponentiel de variables. La complexité de la méthode du simplexe étant liée au nombre de variables, celle-ci ne peut s'exécuter en un temps raisonnable. Dans la méthode de génération de colonnes, seule une partie des variables est générée initialement. Ensuite, un programme linéaire de taille réduite sera utilisé pour générer de nouvelles variables, appelées *colonnes*, jusqu'à obtenir toutes les variables de la solution optimale du problème initial.

Exemple : Soit $\mathcal{R}$ l'ensemble des tournées possibles, d_l le coût de la tournée $l \in \mathcal{R}$, ξ_l une variable binaire indiquant si la tournée est utilisée, $\mathcal{B}_i$ l'ensemble des tournées possibles livrant le client i, V', l'ensemble des clients et m la capacité des véhicules, il est possible de reformuler le modèle linéaire comme indiqué dans la figure 1.10, ci-dessous.

$$
\begin{aligned}
\min \quad & \sum_{l \in \mathcal{R}} d_l \xi_l \\
s.t. \quad & \sum_{l \in \mathcal{B}_i} \xi_l = 1, \quad \forall i \in V' \\
& \sum_{l \in \mathcal{R}} \xi_l \leq m \\
& \xi_l \in \{0,1\} \quad \forall l \in \mathcal{R}
\end{aligned}
$$

FIGURE 1.10 – Modèle du VRP dans sa variante de partitionnement

Dans cette version, modélisée par Balinski et Quandt (1964), les variables sont l'ensemble des chemins réalisables par chacun des véhicules. Le but est donc de choisir parmi ces chemins, ceux qui minimisent la distance totale parcourue. Le nombre de chemins possible étant exponentiel, la génération de colonnes est parfaitement adaptée à cette formulation.

Développée initialement pour des problèmes de découpe (Gilmore et Gomory, 1961), la méthode de génération de colonnes a été améliorée pour fonctionner sur un nombre important de problèmes (Vanderbeck et Wolsey, 1996), avec une quantité toujours croissante de variables (Barnhart *et al.*, 1998).

La génération de colonnes a été adaptée par Skitt et Levary (1985), avant d'être améliorée par Choi et Tcha (2007); Ceselli *et al.* (2009).

Plus de détails sur les mécanismes de la génération de colonnes sont disponibles dans l'article de Feillet (2010).

Méthodes à séparation et évaluation

Suivant l'adage "*Diviser pour mieux régner*", les méthodes à branchement divisent un problème complexe en sous-problèmes plus simples à résoudre. Ces sous-problèmes peuvent également être divisés.

Il existe plusieurs méthodes de branchement, les plus connues sont :

Branch and Bound Cette méthode consiste à partitionner l'espace des solutions. L'espace de recherche initial correspond à un recouvrement de l'espace des solutions réalisables. Cet espace va ensuite être séparé en sous-espaces disjoints. Dans le cas d'un problème de minimisation, deux bornes sont créées pour évaluer les sous-ensembles :

1. La borne supérieure, correspondant à la valeur de la fonction objective de la meilleure solution réalisable obtenue.
2. La borne inférieure, correspondant à une approximation de la plus petite valeur qu'une solution du sous-ensemble peut atteindre. Cette borne peut n'être atteinte par aucune des solutions, mais aucune solution ne peut avoir une valeur inférieure à cette borne.

Les bornes inférieures et supérieures permettent d'évaluer les sous-ensembles : si la borne inférieure d'un sous-ensemble est supérieure à la borne supérieure du problème, aucune des solutions de cet ensemble n'est optimale, et l'ensemble n'est plus étudié.

Les ensembles ne contenant aucune solution réalisable sont également éliminés. L'algorithme divise récursivement les ensembles, jusqu'à ce qu'ils soient tous éliminés. Une des premières versions de l'algorithme de Branch-and-Bound a été développée par Laporte et Nobert (1983), avant d'être adaptée pour le VRP par Fisher (1994).

Branch-and-Cut Le principe est similaire au *Branch-and-Bound* : l'espace des solutions va également être divisé. Cependant, dans le Branch-and-Cut, les sous-ensembles sont résolus via la méthode du simplexe. Régulièrement, des contraintes (appelées *coupes*) sont ajoutées dans les modèles associés aux sous-problèmes. Ces coupes éliminent des solutions non réalisables afin d'améliorer les chances d'obtenir une solution réalisable lors de l'appel du simplex. Cette méthode est donc conçue pour réduire le nombre d'appels à l'algorithme du simplexe.

Branch-and-Price La méthode Branch-and-Price est l'adaptation de la génération de colonnes pour les problèmes contenant des variables entières. Elle combine donc l'accélération de la relaxation linéaire, via la génération de colonnes, et la recherche de solutions entières, via l'algorithme de Branch-and-Bound.

En effet, la génération de colonnes est une méthode pour utiliser l'algorithme du simplexe sans générer toutes les variables.

Branch-and-Cut-and-Price

La méthode de branchement la plus efficace est le Branch-and-Cut-and-Price qui combine les différentes méthodes de branchement. Une variante efficace pour le VRP a été développée par Pessoa *et al.* (2009), avant d'être améliorée par Pecin *et al.* (2017), puis par Pessoa *et al.* (2020). Elle est désormais une méthode de référence et a permis de résoudre de nombreuses instances et variantes difficiles, dont les instances DLP (Duhamel, Lacomme, Prodhon), comme nous le verrons dans le chapitre 3.

Les méthodes exactes sont donc des méthodes garantissant l'optimalité des solutions retournées. Cependant, cette garantie demande énormément de temps avant d'être obtenue. Nous allons donc désormais nous concentrer sur des méthodes sans garantie de résultat, dont le temps d'exécution est beaucoup plus faible.

1.2.2 Méthodes approchées

Comme leur nom l'indique, les méthodes approchées ne garantissent pas d'obtenir la solution optimale. Cependant, leur faible temps d'exécution les rend très attractives dans le milieu industriel. Cette section va présenter les méthodes approchées les plus connues ou efficaces.

Méthodes constructives

Les méthodes constructives permettent de créer une, et une seule solution. Les méthodes constructives utilisent souvent des heuristiques gloutonnes : il s'agit de règles utilisées pour choisir quel client affecter à une tournée, ou le choix d'une arête à activer. Une fois cette affectation effectuée, elle ne sera plus remise en question, réduisant drastiquement les possibilités, et donc le temps d'exécution de cet algorithme.

Plus proche voisin Cette méthode est la plus intuitive : on commence une tournée en livrant le client le plus proche du dépôt. Ensuite, la tournée est complétée en livrant le client le plus proche de cette nouvelle position, et ainsi de suite. Lorsque la demande du nouveau client à affecter est supérieure à la capacité restante du véhicule, le véhicule retourne au dépôt et une nouvelle tournée est créée. L'algorithme se termine lorsque tous les clients ont été affectés à une tournée. Cette méthode est très facile à coder, et rapide à s'exécuter. Cependant, la solution qui en résulte est souvent de mauvaise qualité.

Clarke and Wright Cette méthode est la plus connue et est toujours utilisée. L'algorithme commence par créer une tournée par client. Ensuite, les tournées seront fusionnées jusqu'à ce qu'aucune fusion réduisant le coût de la solution ne soit possible. Soit 0, le dépôt, i le dernier client d'une tournée, j le premier client d'une autre tournée et c_{ij} le coût de l'arête (ij), en fusionnant ces deux tournées, le coût est réduit d'un score (appelé *saving*) $s_{ij} := c_{i0} + c_{0j} - c_{ij}$.

Initialement créée par Clarke et Wright (1964), cette méthode a été généralisée par Gaskell (1967); Yellow (1970), en ajoutant un paramètre λ ($\lambda \geq 0$) déterminant l'importance de la distance au dépôt : $s_{ij} := c_{i0} + c_{0j} - \lambda c_{ij}$. Cette version a elle-même été généralisée pour traiter l'insertion de client au sein d'une tournée Mole et Jameson (1976); Solomon (1987). Un exemple détaillé sur l'exécution de cette méthode est présent dans le chapitre 3.

Méthodes en deux phases

Les problèmes du type VRP sont la fusion de deux problèmes NP-difficiles : le bin packing et le TSP. Certaines méthodes, au lieu de s'occuper des deux aspects de manière simultanée (comme la méthode constructive de Clarke and Wright) s'occupent d'optimiser l'un des sous-problèmes, avant d'optimiser l'autre.

Cluster first, route second Les méthodes du type *cluster first, route second* sont des méthodes qui commencent par regrouper les clients par tournées. C'est seulement une fois les groupes effectués que les tournées seront optimisées comme des problèmes de TSP indépendants. Par exemple, l'algorithme des pétales (Renaud *et al.*, 1996) génère dans un premier temps un nombre important de tournées. Dans un second temps, un problème de partitionnement (figure 1.10) est résolu pour déterminer les tournées à conserver. Une des

méthodes les plus connues et anciennes (Gillett et Miller, 1974) est appelée *sweep algorithm* et regroupe les clients par coordonnées polaires, avant d'optimiser indépendamment chaque tournée.

Route first cluster second Ces méthodes commencent par minimiser les temps de trajet, avant d'affecter les clients aux tournées. Ce type de méthode s'est vraiment développé à partir de l'utilisation de la méthode SPLIT (Beasley, 1983) qui permet de transformer une solution du TSP en solution pour le VRP.

En effet, de nombreuses métaheuristiques récentes créent en premier lieu des solutions composées d'une seule tournée (appelée tour géant, ou *giant tour*), avant de la découper pour connaître le coût réel de la solution non relaxée. Elle est en particulier utilisée par les méthodes génétiques (Prins, 2004; Vidal *et al.*, 2014), où coder une solution comme une simple permutation facilite l'utilisation de méthodes de crossover.

Plus d'informations sur ces méthodes, et en particulier sur le changement de codage de la solution sont disponibles dans l'état de l'art écrit par Prins *et al.* (2014).

Interaction entre les phases Une fois les deux phases effectuées, il est toujours possible de vérifier si le sous-problème initial peut à nouveau être optimisé. Dans un article récent Toffolo *et al.* (2019) ont associé à chaque partitionnement, une solution de bonne qualité. Ainsi, les auteurs alternent entre recherche d'un bon partitionnement et recherche du meilleur trajet associé.

Méthodes à recherche locale

Les solutions créées à partir d'un algorithme constructif sont obtenues rapidement, mais sont généralement de mauvaise qualité. Souvent, quelques perturbations suffisent pour améliorer une solution. Les méthodes à *recherche locale* étudient une ou plusieurs familles de perturbations de la solution courante. L'ensemble des solutions accessibles via une perturbation est appelé *voisinage*. Les perturbations sont appelées *mouvement*, mais sont parfois nommé *voisinage*, par abus de langage. Ces méthodes s'arrêtent lorsque aucune solution du voisinage n'est meilleure que la solution courante. On dit alors que la recherche locale a atteint un minimum local. Cette partie constitue le cœur des solveurs implémentés et sera expliquée plus en détail dans le chapitre 3.

Voisinages fréquents

Bien que les possibilités de modifications soient importantes, la plupart des méthodes à recherche locale utilisent les mêmes voisinages.

Il existe principalement deux façons de modifier une solution. La plus connue consiste à retirer k arêtes d'une solution et à les remplacer par k autres arêtes. Ce mouvement est intitulé k-opt. Le temps d'exécution augmentant avec le paramètre k, le k-opt le plus utilisé est le 2-opt, très utile pour réduire le nombre de croisements, aussi bien entre tournées, qu'au sein d'une tournée.

Une variante des voisinages k-opt est l'algorithme de Lin et Kernighan (1973) qui étudie plusieurs paramètres k, jusqu'à obtenir une bonne solution. Les solutions obtenues sont généralement de bonne qualité, mais l'implémentation est plus longue et difficile, et le temps est plus élevé, malgré l'amélioration des performances par Helsgaun (2000).

Une autre façon consiste à échanger les positions et affectations de α clients d'une tournée et de β clients d'une seconde tournée. Lorsque les valeurs de α et β sont faibles, les voisinages associés ont été utilisés depuis des décennies et ont leur propre nom. Ainsi, lorsque $\alpha = 1$ et $\beta = 0$, le voisinage résultant est nommé *réallocation*. Si $\alpha = 1$ et $\beta = 1$, le voisinage se nomme *swap exchange*, et ainsi de suite.

Ces voisinages sont utilisés depuis des décennies, et sont notamment listés dans les articles de Van Breedam (1994) ou Kindervater et Savelsbergh (1997). La plupart de ces voisinages sont décrits dans le chapitre 3 lors de la description du solveur implémenté.

Adaptive Large Neighborhood Search (ALNS)

Certains voisinages fonctionnent en deux étapes : la solution est initialement en partie détruite (par exemple, en supprimant des arêtes, ou en supprimant l'affectation de certains clients), avant d'être réparée dans une seconde étape. Ces voisinages sont appelés *voisinages larges*, ou *larges neighborhood*.

Comme leur nom le suggère, ces voisinages sont de taille conséquente, et nécessitent un temps d'exécution important. C'est pour éviter d'utiliser aléatoirement ces voisinages, et ainsi prendre le risque de perdre un temps précieux à faire une recherche locale avec un voisinage peu utile que la méthode *Adaptive Large Neighborhood Search* a été inventée.

Dans cette méthode, plusieurs voisinages larges sont utilisés. Initialement, les voisinages sont utilisés à la même fréquence. Cependant, au fil des itérations, la fréquence d'utilisation des voisinages ayant peu servi est réduite. Inversement, un voisinage ayant

fait ses preuves sera appelé plus souvent.

Inspirée du recuit simulé (Van Laarhoven et Aarts, 1987), et inventée par Schrimpf *et al.* (2000), cette méthode a vite été adaptée aux problèmes de tournées (Ropke et Pisinger, 2006), améliorée (Masson *et al.*, 2013; Dayarian *et al.*, 2016), et adaptée pour fonctionner sur plusieurs variantes du VRP (Grangier *et al.*, 2016). Cette méthode continue d'être une source d'inspiration pour des travaux récents sur le VRP (Christiaens et Vanden Berghe, 2020).

Amélioration des voisinages

Les méthodes à recherches locales ont des temps d'exécution très élevés. Ainsi, il est toujours intéressant de voir s'il est possible de réduire la taille du voisinage, afin de réduire les calculs inutiles.

La manière la plus connue consiste à trier les arêtes partant du client i, et à ne conserver que les k plus courtes (Johnson et McGeoch, 1997). Le nombre d'arêtes passe ainsi de n^2 à $k * n$. Le choix du paramètre k a été étudié par Arnold *et al.* (2019a).

Une variante, suggérée par Toth et Vigo (2003) consiste à ne conserver que les arêtes dont la taille ne dépasse pas $\beta\ \overline{z}$, où β est un paramètre, et $\overline{z}$ la taille moyenne des arêtes. L'idée est similaire, il s'agit de réduire l'espace de recherche en n'étudiant que les solutions qui semblent intéressantes.

Greedy Randomized Adaptive Search Procedure (GRASP)

Les méthodes constructives étant déterministes, elles ne donnent qu'une seule solution. La méthode GRASP (*Greedy Randomized Adaptive Search Procedure*) permet d'ajouter de l'aléatoire dans une méthode constructive. Dans l'algorithme de Clarke and Wright, expliqué précédemment, un score s_{ij}, appelé *saving* est calculé. À chaque itération, le couple (i, j) éligible avec le plus haut saving, que l'on notera $\overline{s}$ est choisi. Dans la version randomisée, le couple (i, j) est choisi aléatoirement parmi ceux vérifiant $s_{ij} \geq \alpha\ \overline{s}$, où le paramètre $\alpha\ \in [0; 1]$.

L'idée de la méthode GRASP apparaît pour la première fois pour des problèmes de recouvrement (Feo et Resende, 1989), mais l'idée d'ajouter de l'aléatoire commençait à apparaître (Hart et Shogan, 1987). La méthode GRASP a été généralisée pour toutes les méthodes constructives par Feo et Resende (1995). Dans cet article, la méthode est constituée d'une seconde phase, où les solutions obtenues sont améliorées par recherche

locale. Cette méthode est désormais populaire et sert désormais de base pour des solutions initiales.

La recherche tabou

Au lieu de créer une nouvelle solution lorsqu'un minimum local a été atteint, il est possible de continuer la recherche. La méthode de recherche tabou (ou *Tabu Search*), choisit toujours la meilleure solution du voisinage, même si celle-ci est moins intéressante que la solution courante. Pour éviter de rester bloqué sur les mêmes solutions, une liste de mouvements ou solutions taboues est régulièrement mise à jour, forçant l'exploration de nouvelles solutions.

La méthode de recherche tabou a été développée par Glover (1986), avant d'être adaptée pour le problème de tournées (Gendreau *et al.*, 1994), et ses variantes (Cordeau *et al.*, 2001). Plus de détails sur la méthode sont disponibles sur l'état de l'art écrit par Glover et Laguna (1998).

Méthodes à base de population

Au lieu de se baser sur une solution à améliorer, certaines méthodes se basent sur un ensemble de solutions. Ces méthodes à bases de population (ou *population based methods*) sont très populaires et font partie des méthodes les plus efficaces.

Algorithme de colonies de fourmis

Inventé par Dorigo et Di Caro (1999) pour le TSP, l'algorithme de *Ant Colony Optimisation* (ACO) s'inspire du comportement des fourmis : les fourmis ayant trouvé un chemin menant à de la nourriture vont émettre des phéromones, incitant les fourmis suivantes à suivre ce chemin. Pour le TSP, un ensemble de solutions, appelées *fourmis* va être généré aléatoirement. Les solutions les plus intéressantes vont laisser des phéromones sur les arêtes utilisées. La création de la deuxième génération de fourmis sera ensuite influencée par les phéromones, créant des solutions, toujours aléatoires, mais guidées par les bonnes solutions obtenues précédemment. Plusieurs générations de fourmis vont ensuite se succéder jusqu'à obtention du critère d'arrêt. Cette méthode a vite été adaptée pour le VRP (Bullnheimer *et al.*, 1999). Un article détaillé sur le fonctionnement de l'ACO a été créé par Dorigo *et al.* (2006).

Algorithmes génétiques

La métaheuristique à base de population la plus connue, générique et efficace est l'algorithme génétique. Après avoir créé une population initiale d'individus, une nouvelle génération va être créée à partir de l'ancienne. La façon la plus répandue consiste à choisir aléatoirement deux individus et à fabriquer plusieurs solutions en se basant sur les solutions parentes. Ensuite, une partie de ces nouvelles solutions va subir une mutation : une perturbation aléatoire va leur être appliquée. Enfin, un filtrage va être effectué afin de garder essentiellement les meilleures solutions.

Cette métaheuristique est utilisée pour la première fois par Holland (1975) et rendu populaire par Goldberg (2006). Une première version efficace de l'algorithme génétique pour le VRP a été implémentée par Prins (2004), puis améliorée par Vidal *et al.* (2012), qui l'a ensuite généralisée pour la plupart des variantes du VRP (Vidal *et al.*, 2014).

1.3 Prise en compte des flottes hétérogènes

1.3.1 Liste des problèmes hétérogènes

Comme étudié précédemment, il existe des variantes du VRP prenant en compte les différents types de véhicules. Ces problèmes de flottes hétérogènes ont également plusieurs variantes, classées par Baldacci *et al.* (2008) selon trois critères :

La taille des flottes Certaines instances estiment que le nombre de véhicules de chaque catégorie est suffisamment important pour être considéré comme infini, on parle alors de *Fleet Size and Mix VRP* (FSM). Dans le cas contraire, on parle *Heterogeneous VRP* (HVRP).

Les coûts fixes Dans cette variante, utiliser un véhicule de catégorie k ajoute un coût F_k indépendant du trajet effectué. On parle alors de HVRP / FSM *with Fleet Cost* (HVRPF / FSMF).

Les coûts d'itinéraire Des véhicules différents n'ont pas toujours la même consommation pour la même distance parcourue. De plus, si le coût horaire est pris en compte, la différence de vitesse entre les véhicules peut impacter le coût final de la tournée. Dans cette variante, le graphe des distances est dépendant du véhicule utilisé (voir figure 1.6).

Dans la majorité des instances de la littérature, les graphes sont liés par un coefficient de vitesse : Soit d_{ij}^k le coût pour parcourir l'arête (i, j) avec un véhicule de type k, et v_k le coefficient de vitesse, on a : $d_{ij}^k = v_k \; d_{ij}^1$. Lorsque coût de l'arête (i, j) est dépendant du véhicule, on parle alors de HVRP / FSM *with Vehicle Dependent Routing Costs* (HVRPD / FSMD).

En prenant en compte l'ensemble des possibilités, on obtient donc les différentes appellations, résumées dans le tableau 1.1.

Nom	Taille de la flotte	Coûts Fixes	Coûts d'itinéraires
HVRPFD	Limitée	> 0	Dépendant du véhicule
HVRPD	Limitée	$= 0$	Dépendant du véhicule
FSMFD	Illimitée	> 0	Dépendant du véhicule
FSMD	Illimitée	$= 0$	Dépendant du véhicule
FSMF	Illimitée	> 0	Indépendant du véhicule

Tableau 1.1 – Récapitulatif des variantes avec flottes hétérogènes

Quelle que soit la variante traitée, la capacité du véhicule est toujours considérée comme dépendante de la catégorie utilisée.

Lorsque la variante du véhicule hétérogène n'est pas précisée, nous parlerons de *problèmes de tournées avec flotte hétérogène*, ou HVRP.

1.3.2 État de l'art concernant les flottes hétérogènes

Les premiers problèmes hétérogènes ont été proposés par Golden *et al.* (1984), dans leur version la plus simple (FSMF). Les contraintes de limite de véhicules et vitesses hétérogènes ont été ajoutées dans les années qui ont suivi, jusqu'à obtenir des problèmes avec toutes les contraintes (HVRPFD), proposés par Li *et al.* (2007). Les autres variantes du VRP ont aussi été adaptées pour les flottes hétérogènes, comme l'apparition de problèmes hétérogènes avec fenêtre de temps, par Liu et Shen (1999).

Afin de prendre en compte la difficulté de ces instances, certaines nouvelles techniques ont été inventées. Ainsi, de nouveaux voisinages, cherchant à modifier les tournées utilisées, ont été créés par Salhi et Rand (1993). Concernant les méthodes exactes, de nouvelles coupes ont été ajoutées par Yaman (2006).

Cependant, la plupart des articles de la littérature sont constitués de méthodes déjà existantes, mais adaptées pour les flottes hétérogènes. Ainsi, le calcul des savings a été adapté par Golden *et al.* (1984), puis modifié par Gheysens *et al.* (1986) et Desrochers et Verhoog (1991) pour prendre en compte les différentes variantes. Plus récemment, Juan *et al.* (2014) ont adapté les méthodes constructives en résolvant initialement un problème homogène, puis en enlevant les tournées excédentaires, pour appeler à nouveau l'algorithme constructif sur les clients restants, en utilisant une autre catégorie de véhicule. L'algorithme constructif *sweep* a également été adapté (Renaud et Boctor, 2002).

Les métaheuristiques classiques ont également été adaptées, comme la recherche tabou (Gendreau *et al.*, 1999; Brandão, 2011), ou les variantes de l'algorithme génétique (Prins, 2009), et du recuit simulé (Li *et al.*, 2007). L'algorithme de génération de colonnes a également été adapté en version heuristique par Taillard (1999), où des chemins pour des problèmes homogènes ont été créés, avant de chercher ceux à activer.

L'article de référence pour les méthodes exactes des problèmes hétérogènes est celui de Pecin *et al.* (2017) où la méthode de Branch-and-Cut-and-Price permet de résoudre un nombre important d'instances difficiles de la littérature.

Enfin, la métaheuristique de référence est l'algorithme génétique développé par Vidal *et al.* (2014) et compatible avec de nombreuses variantes du VRP.

Il est intéressant de noter que cet algorithme est basé sur des solutions représentées par des tours géants, avant de les diviser par l'algorithme SPLIT, comme suggéré par Golden *et al.* (1984), 30 ans plus tôt.

L'algorithme SPLIT, développé par Beasley (1983) permet de déterminer la meilleure affectation clients / véhicules pour un ordre de livraison fixé. Cette méthode est donc presque indispensable pour résoudre les instances hétérogènes, où une mauvaise affectation a un impact majeur sur le coût de la solution. La complexité est polynomiale, voire linéaire (Vidal, 2016) si la flotte est de taille illimitée. Dans le cas contraire, des variantes non polynomiales ont été créées (Duhamel *et al.*, 2010).

Les problèmes hétérogènes ont été étudiés depuis plus de 30 ans, il existe donc plusieurs articles listant l'état de l'art sur ces problèmes, ainsi que les différentes variantes existantes (Baldacci *et al.*, 2008; Irnich *et al.*, 2014; Koç *et al.*, 2016).

1.4 Adaptation à la mobilité urbaine

En milieu urbain, avec les sens uniques, l'imprévisibilité du trafic, la difficulté pour stationner et les zones réservées aux piétons, il est parfois difficile de circuler.

Dans cette section, nous allons étudier une partie des problématiques liées à la ville, les défis qui sont posés par la recherche opérationnelle, et les réponses qui ont été apportées.

1.4.1 Réduction de la pollution en milieu urbain

Un des premiers points importants concernant la mobilité urbaine, est la pollution. En effet, le trafic routier est en partie responsable de la pollution dans les villes (Gühnemann *et al.*, 2004). C'est dans ce contexte qu'est apparu le problème de *Green VRP* (GVRP), où des contraintes limitant l'émission totale de CO2 sont ajoutées au problème initial (Erdoğan et Miller-Hooks, 2012).

Rapidement, de nombreuses variantes sont apparues, comme l'idée de résoudre un problème bi-objectif, minimisant à la fois la distance parcourue et la quantité de CO2 émise (Jemai *et al.*, 2012). Dans le but de réduire la pollution, certains articles étudient l'impact de l'utilisation de véhicules électriques, ou de flottes hétérogènes sur la pollution Davis et Figliozzi (2013); Moutaoukil *et al.* (2014). Un état de l'art des problèmes GVRP a été réalisé par Lin *et al.* (2014).

1.4.2 Les problèmes à niveaux multiples

Étant donné la difficulté de circulation dans les zones denses, il est rare que les véhicules imposants prennent part à la livraison en centre-ville. Il est donc fréquent que la livraison se déroule en deux étapes : les véhicules à forte capacité s'occupent de la livraison entre les villes et déposent leurs contenus dans des zones d'échange spécifiques appelées *zones de distribution*. Enfin, des véhicules plus légers s'occupent de la livraison au sein de chaque ville.

L'idée de prendre en compte les centres de distributions a été soutenue par Crainic *et al.* (2004), ce qui a mené à l'apparition des problèmes à deux échelons (Crainic *et al.*, 2010). Dans cette première version, deux catégories de véhicules sont pris en compte : ceux servant uniquement à la livraison du dépôt principal aux centre de distributions, et ceux partant des centres des distributions et livrant les clients.

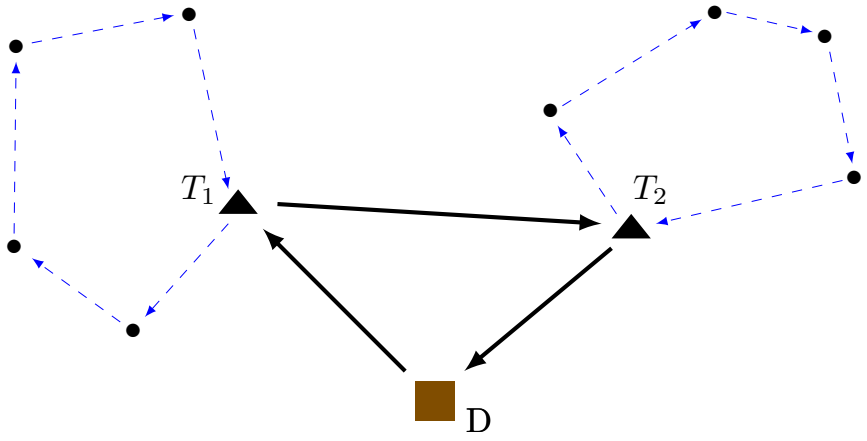


FIGURE 1.11 – Exemple de problème à 2 échelons

De nombreuses variantes sont apparues, dont les problèmes de synchronisation Grangier *et al.* (2016), où les produits déchargés de la première catégorie de véhicule sont ensuite chargés directement dans la deuxième catégorie, nécessitant que ceux-ci soient présents au même moment sur le même centre de distribution. Plus d'informations sur ce type de problème sont disponibles dans l'état de l'art écrit par Cuda *et al.* (2015).

1.4.3 Variation de la congestion

La vitesse de circulation en ville est très instable en fonction du temps. Dans leurs travaux, Ehmke *et al.* (2010) ont modélisé quatre périodes aux vitesses de circulations différentes : la nuit, le début de journée, le milieu de journée et la fin de journée. Les auteurs ont réussi à montrer que prendre en compte les différents intervalles de temps, au lieu d'une vitesse moyenne, permet une réduction jusqu'à 20% du temps de trajet (Ehmke et Mattfeld, 2010; Ehmke *et al.*, 2012).

Ces problèmes, dit *Time Dependent*, (déjà évoqués en section 1.1.3) commencent à être fortement étudiés. En effet, la multiplication des données publiques, et des algorithmes permettant de faciliter leur compréhension permet de prédire efficacement la congestion en fonction de l'heure (Tsubota *et al.*, 2011). Ainsi, Melgarejo *et al.* (2015) ont utilisé les données de la ville de Lyon pour créer des instances réalistes dépendant du temps.

Cependant, en pratique, il est difficile de prédire avec précision l'évolution du trafic. Une première manière de s'adapter est d'anticiper plusieurs scénarios et de résoudre un problème où les données sont représentées par la probabilité de distributions entre les différents scénarios possibles, on parle alors de problèmes *stochastiques*. Il est possible

également, de traiter des problèmes où la demande est également probabiliste (Saint-Guillain *et al.*, 2017). Une autre approche consiste à étudier régulièrement les données, et à ajuster la solution pour tenir compte des dernières informations Dans la littérature, ces variantes sont nommées *Dynamiques* VRP (DVRP).

Pour plus d'informations sur ces variantes, des articles d'état de l'art sont disponibles (Pillac *et al.*, 2013; Ritzinger *et al.*, 2016).

1.4.4 Pour aller plus loin

Dans cette section, nous avons vu une partie des problématiques liées aux milieux urbains, et aux variantes du VRP qui en résultent. Il en existe de nombreuses autres, comme l'affectation de véhicules à certains quartiers (Franceschetti *et al.*, 2017), ou les problèmes d'*Access Time windows* (Muñuzuri *et al.*, 2013), où certaines zones ne sont accessibles que sur un intervalle de temps donné.

Plus d'informations sur les problématiques de la mobilité urbaine sont disponibles dans les articles de Ambrosini et Routhier (2004) et Anderson *et al.* (2005). Enfin, Cattaruzza *et al.* (2017) nous proposent un état de l'art sur plusieurs variantes du VRP liées à la mobilité urbaine.

Ainsi, la livraison en milieu urbain et les flottes hétérogènes sont deux problèmes majeurs de la mobilité actuelle et future. Dans les chapitres suivants, nous étudierons des problèmes combinant ces deux particularités : la livraison en milieu urbain, avec plusieurs catégories de véhicules, qui devront s'adapter aux zones de différentes densités.

1.5 Performances des méthodes de résolution

Un article indispensable pour avoir une idée claire sur les performances des algorithmes actuels est écrit par Uchoa *et al.* (2017) et compare les performances et temps de calcul des meilleurs algorithmes approchés et exacts, connus au moment de la rédaction de l'article. Un nouveau jeu d'instances, contenant 100 instances de 100 à 1000 clients a été créé. Ce jeu a été testé sur trois algorithmes de références :

— L'algorithme ILS-SP (Subramanian *et al.*, 2013), basé sur des recherches locales. Chaque nouvelle recherche se base sur les informations liées aux minima locaux obtenus précédemment. Chaque fois qu'un minimum local a été atteint, la solution courante subit une perturbation, avant d'appliquer de nouveau une recherche locale.

— L'algorithme UHGS (Vidal *et al.*, 2012, 2014) qui est basé sur les méthodes génétiques, avec la particularité de toujours forcer à avoir une base de population aussi variée que possible. De plus, cette méthode s'adapte parfaitement à de nombreuses variantes du VRP.

— L'algorithme de Branch-and-Cut-and-Price (Pecin *et al.*, 2017) qui réussit à regrouper toutes les idées proposées quelques années avant. C'est le seul algorithme parmi les trois testés qui est exact.

L'algorithme de Branch-and-Cut-and-Price est capable de résoudre de manière exacte 43 des 100 instances testées, (quasi exclusivement celles de taille inférieure à 300 clients), et trouve des solutions ayant un coût moins de 1% supérieur à la borne inférieure pour 96 instances. Les solutions obtenues par cette méthode sont donc de très bonne qualité. Cependant, il faut souvent un à plusieurs jours de calculs pour obtenir ces résultats, ce qui est rarement utilisable en pratique.

Les algorithmes ILS-SP et UHGS qui sont des méthodes approchées obtiennent des solutions beaucoup plus rapidement. L'algorithme UHGS est celui donnant le plus fréquemment les solutions de meilleure qualité, et le plus rapidement. Il obtient en moyenne un écart de 0.52% par rapport à la borne inférieure. Les solutions sont donc de très bonne qualité. Concernant les temps de calculs, les instances comportant entre 100 et 200 clients sont généralement résolues en quelques minutes, celles comportant environ 500 clients sont résolues en une heure et celles comportant 1000 clients sont résolues en une dizaine d'heures.

Réduction du temps de calcul Le temps d'exécution pour obtenir des solutions augmente donc de manière conséquente avec la taille des instances. Dans leur article étudiant la taille optimale des graphes pour chaque voisinage lors d'une recherche locale, Arnold *et al.* (2019a) ont réduit significativement les temps de calculs nécessaires. En effet, ils sont passés de voisinages en $O(n^2)$, voire plus, à des voisinages en $O(k * n)$, où k est un paramètre. Dans leur étude, les auteurs ont fixé ce paramètre à 30, ou 50, selon le voisinage à exécuter. Avec cette réduction de graphe, il est désormais possible d'obtenir des solutions de bonne qualité en quelques minutes sur des instances contenant plusieurs milliers de clients, contre plus d'une dizaine d'heures avec un graphe complet.

1.6 Conclusion

Il existe plusieurs milliers de publications sur les tournées de véhicules. Ce chapitre ne contient donc qu'un échantillon des travaux effectués par la communauté scientifique. Heureusement, des synthèses sur les publications concernant les tournées de véhicules sont faites très régulièrement, permettant d'avoir rapidement un aperçu de la littérature à une date donnée. Concernant les problèmes généraux de tournées de véhicules, les publications suivantes permettent de bien étudier les travaux déjà effectués : Laporte (1992); Cordeau *et al.* (2007); Golden *et al.* (2008); Laporte (2009); Golden *et al.* (2008); Toth et Vigo (2014); Braekers *et al.* (2016); Vidal *et al.* (2019).

Les problèmes avec flottes hétérogènes étant particulièrement étudiés, des recueils de publications leur ont été dédiés. Ainsi, Baldacci *et al.* (2008) ont écrit l'état de l'art des modèles mathématiques, bornes inférieures et heuristiques liées aux problèmes hétérogènes. Ce recueil a été complété par un inventaire des méthodes exactes, surtout axé sur les coupes et les méthodes de séparation et évaluation (Baldacci *et al.*, 2010). De plus, une mise à jour des méthodes utilisées a été publiée quelques années plus tard Irnich *et al.* (2014).

Enfin, la publication de Vidal *et al.* (2013) permet de mieux identifier les méthodes utilisées selon les combinaisons de variantes à résoudre.

Chapitre 2

Machine learning pour la recherche opérationnelle

Dans cette section, nous allons voir les interactions possibles entre apprentissage automatique et recherche opérationnelle. Nous allons commencer par faire un bref inventaire des méthodes d'apprentissage. Ensuite, nous allons étudier les similitudes entre ces deux domaines. Enfin, nous allons voir comment ces disciplines peuvent être combinées, pour améliorer les performances des méthodes de recherche opérationnelle.

Sommaire

2.1 Brève revue des méthodes d'apprentissage

Dans cette section, nous présenterons les différents domaines de l'apprentissage statistique, et les principales méthodes associées.

2.1.1 Les méthodes supervisées

Il s'agit de la branche la plus connue de l'apprentissage automatique. Ici, nous disposons d'un ensemble de données, pour lesquelles l'un des attributs représente une catégorie à prédire, dite aussi classe, label, étiquette, ou cible. Pour une partie des données, appelée ensemble d'apprentissage (*training set*), les classes sont connues. Le but des méthodes supervisées est de créer des règles de décision permettant de prédire la classe de chacune des données à partir de cet ensemble d'apprentissage. La règle de décision construite, ou classifieur, est ensuite testée sur un ensemble de test (*test set*), sous-ensemble de données pour lequel les classes sont d'abord supposées inconnues, et qui servira à évaluer les performances de la méthode. Sans perte de généralité, nous supposerons dans cette section que les données sont séparées en exactement deux classes.

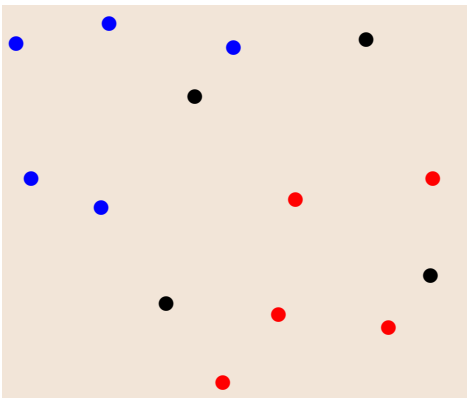


FIGURE 2.1 – Exemple de données d'entraînement et de test

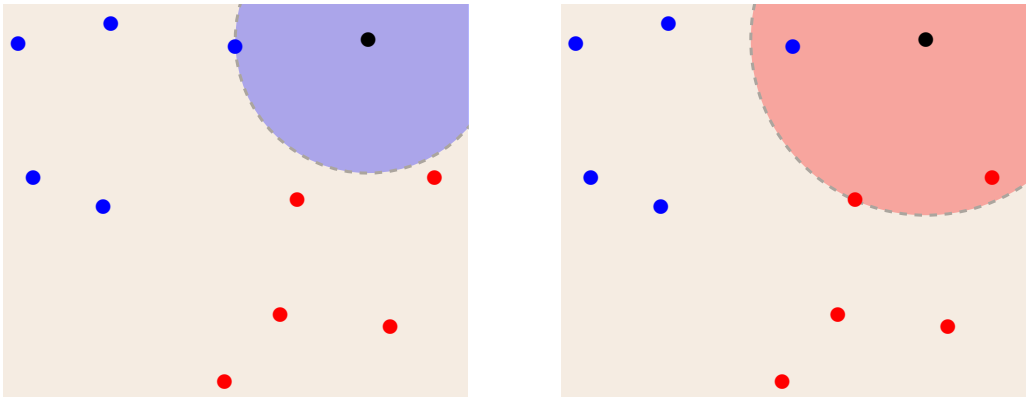
Par exemple, dans la figure 2.1, les points en bleu correspondent aux données d'apprentissage de la première classe, celles en rouge sont celles de la seconde classe. Enfin, les données en noir représentent l'ensemble de test, dont on suppose ne pas connaître la classe associée. Quelle que soit la méthode utilisée, les règles de classification établies sont dépendantes des données disponibles pendant la phase d'apprentissage. Afin d'améliorer la robustesse de la classification, Kohavi et al. (1995) ont développé la validation croisée.

Il s'agit ici d'entraîner et de tester le modèle sur plusieurs sous-ensembles de données d'apprentissage afin d'obtenir un modèle plus robuste, en matière de biais et de variance, et de limiter les phénomènes de sur-apprentissage (situation où les règles sont dépendantes des spécificités des données de l'ensemble d'entraînement et ne correspondent pas aux données extérieur à cet ensemble).

Dans cette section, nous allons lister quelques méthodes d'apprentissage automatique. La liste est volontairement réduite, nous nous limiterons ainsi aux méthodes les plus présentes dans la littérature mêlant optimisation et apprentissage et à celles mobilisées pour la suite de nos travaux.

Les k plus proches voisins

Il est très intuitif de supposer qu'une donnée de classe i sera proche des données également de classe i. La méthode des k plus proches voisins, ou k nearest neighbors (KNN) exploite cette intuition. Pour chaque donnée à classer, la méthode va étudier les k données les plus proches (où k est un paramètre) et affecter à la donnée à étudier la classe majoritaire parmi les données voisines.



(a) $k = 1$ (b) $k = 3$

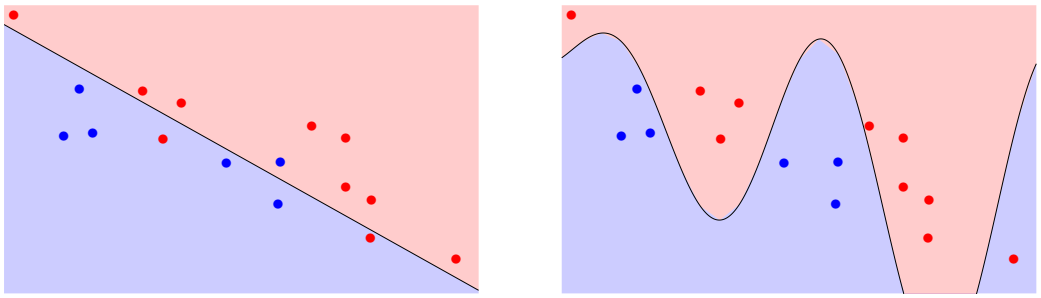
FIGURE 2.2 – Exemple de KNN

Classificateurs linéaires

Les méthodes à classification linéaire cherchent un hyperplan séparant l'espace des données $\mathcal{E}$ en deux sous-espaces $\mathcal{E}_1$ et $\mathcal{E}_2$ contenant respectivement (quasi-)uniquement les données des classes 1 et 2. Ces méthodes sont issues du domaine de l'algèbre linéaire, ce qui permet de bénéficier de propriétés intéressantes.

Cependant, en pratique, il est souvent impossible de séparer les classes par un hyperplan (voir figures 2.3(a) et 2.4(a)). Pour contourner le problème, il existe plusieurs solutions :

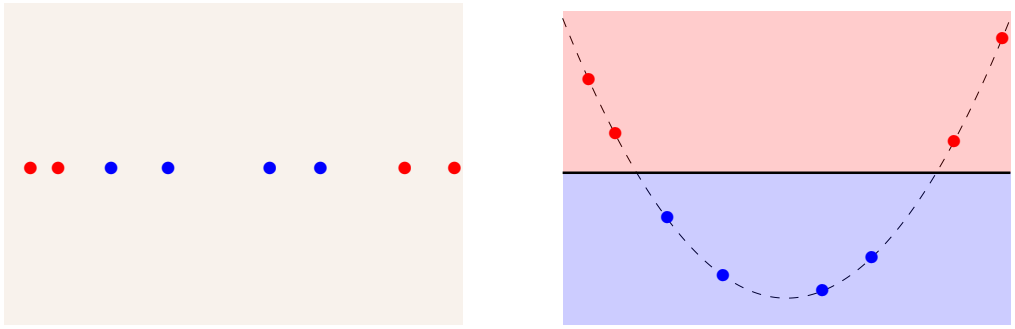
- Effectuer un changement de variable non linéaire. Ainsi, l'hyperplan peut s'adapter à la topologie des données d'entraînement.



(a) Séparation linéaire (b) Transformation non linéaire

FIGURE 2.3 – Exemple de transformation non linéaire

- Ajouter des dimensions. La transformation non linéaire peut aussi modifier la dimension de l'espace d'arrivée. Cette nouvelle dimension peut ainsi faire partie intégrante de l'hyperplan de séparation.



(a) (b)

FIGURE 2.4 – Exemple d'ajout de dimension

Régression linéaire Cette méthode essaie de créer l'hyperplan qui réduit la valeur d'une fonction, appelée *fonction d'erreur*. Selon les variantes, cette fonction peut correspondre au nombre de solutions mal classées, ou à la somme des distances entre les solutions mal classées et l'hyperplan séparateur.

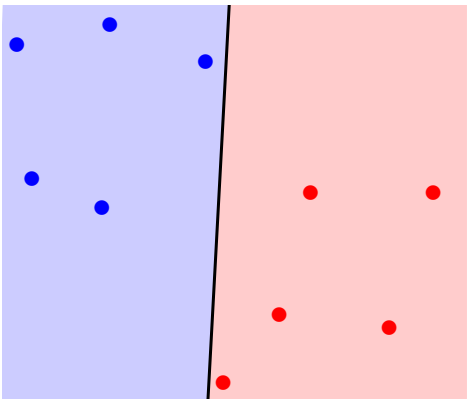


FIGURE 2.5 – Exemple de régression linéaire

Dans la figure 2.5, les solutions à gauche de l'hyperplan sont supposées de catégorie 1, et celle à droite de l'hyperplan sont classées dans la deuxième catégorie.

Machine à vecteurs de supports (SVM) Il existe souvent une infinité d'hyperplans permettant de minimiser la fonction d'erreur, décrite plus haut. Cependant, certains sont plus robustes que d'autres. Celui de la figure 2.5, bien que permettant de classer correctement la totalité de données d'entraînement semble peu robuste. En effet, l'hyperplan est proche des solutions d'entraînement, ce qui signifie que des solutions de test proches des données de la classe 1 seront classées dans la classe 2, et inversement.

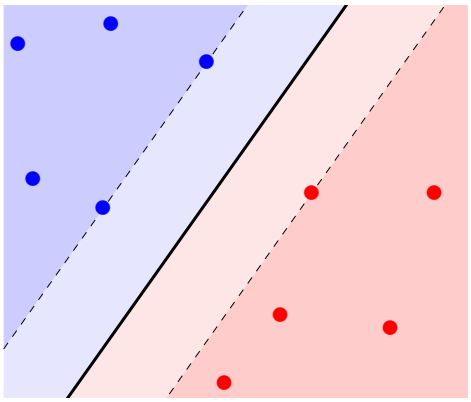


FIGURE 2.6 – Exemple de SVM

Dans la méthode de machine à vecteurs de supports ou *Support Vector Machine* (SVM), le but est de choisir l'hyperplan maximisant la distance avec les données, parmi les hyperplans minimisant la fonction d'erreur.

Dans la figure 2.6, l'hyperplan obtenu minimise les risques de classification erronée.

Les modèles neuronaux

Une autre manière d'apprendre s'inspire de l'organisation des neurones dans le cerveau humain. Initialement basées sur le fonctionnement d'un seul neurone, les méthodes les plus récentes se fondent sur plusieurs milliers de neurones interagissant entre eux, voir beaucoup plus.

Perceptron Inventé par Rosenblatt (1958), le perceptron est le premier modèle utilisant un neurone. Il s'agit d'une entité qui reçoit un ensemble d'informations en entrée et qui va la convertir en sortie après application d'un certain type de fonction.

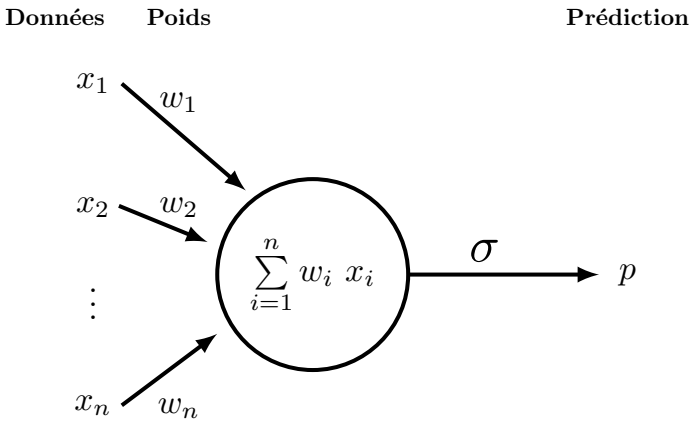
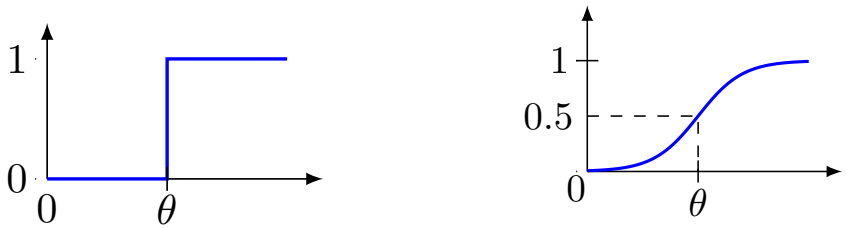


FIGURE 2.7 – Représentation d'un neurone seul

Soit x_i les informations reçues en entrée, le perceptron va calculer la fonction suivante : $f(x) = \sum_{i=1}^{n} x_i * w_i$, où w_i correspond au poids de chaque information.

La fonction est ensuite comparée à une valeur seuil θ. Si $f(x)$ est inférieur, le perceptron va supposer que la donnée testée est de la première classe, sinon, il va supposer qu'elle est de la deuxième (voir figure 2.8(a)).

Cependant, cette façon d'agir ne permet pas de connaître les nuances sur la certitude de la classe de la donnée de test. Pour améliorer ce point, la fonction de classification est souvent lissée, ce qui donne une fonction sigmoïde (voir figure 2.8(b)), on parle alors de *régression logistique*. La sigmoïde a l'avantage d'indiquer si la donnée se trouve proche de la frontière entre les deux classes.



(a) Classification binaire (b) Exemple de sigmoïde

FIGURE 2.8 – Exemple d'ajout de dimension

Pendant la phase d'entraînement, les poids w_i sont ajustés pour que la prédiction soit aussi proche que possible des observations.

Réseaux de neurones Les règles créées par le perceptron sont cependant beaucoup trop simples pour expliquer la plupart des données réelles. Pour ajouter plus de complexité dans les règles, il est courant d'ajouter plusieurs neurones, qui interagiront entre eux. Cependant, le principe général reste similaire au perceptron.

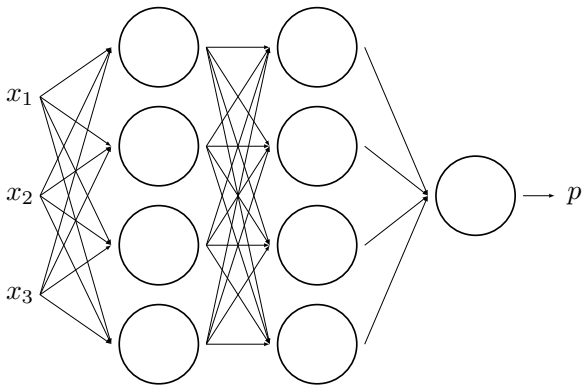


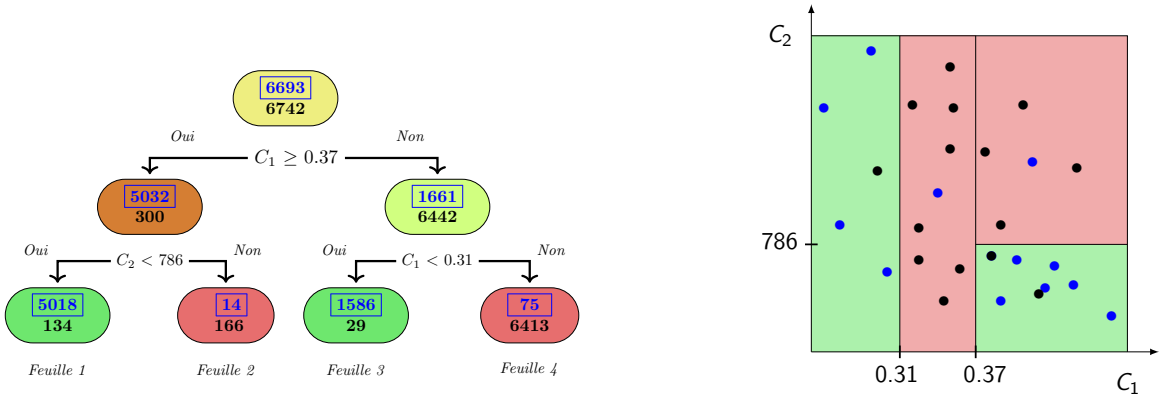
FIGURE 2.9 – Exemple de réseau de neurones

Un réseau de neurones avec un nombre important de couches (appelé *réseau de neurones profond*) a besoin de beaucoup de données pour être efficace. Cependant, pour des problèmes très complexes, avec une base d'apprentissage très conséquente, les réseaux de neurones profonds sont extrêmement performants. Avec l'essor des données ouvertes, les réseaux de neurones disposent désormais de bases d'apprentissage gigantesques et sont largement utilisés dans de nombreux domaines. Il existe donc de nombreux états de l'art recensant les variantes utilisées et leur domaine d'application (Liu *et al.*, 2017; Abiodun *et al.*, 2018). Le livre de Goodfellow *et al.* (2016) est une référence pour comprendre en détail les réseaux de neurones.

Malgré leur haute performance, les réseaux de neurones font partie des algorithmes dits *boîte noire*, c'est-à-dire qu'il est souvent impossible de comprendre les règles de décision qui ont été créées. Cet algorithme ne peut donc permettre une meilleure compréhension des données, et ne peut être utilisé lorsque la prédiction doit être expliquée.

Arbres et forêts

Arbres de décision Les méthodes à arbres de décision partitionnent de manière récursive l'espace des données en plusieurs sous-espaces, en sélectionnant automatiquement à chaque étape la variable la plus discriminante pour une séparation des données, selon un certain critère. À chaque étape, un des sous-espaces est lui-même partitionné jusqu'à ce que les sous-espaces obtenus soient tous considérés comme homogènes. Chaque étape de partitionnement ajoute un hyperplan défini uniquement par une dimension (par exemple : si $D_1 \leq 1.5$ les données sont classées dans la catégorie 1. Sinon, elles sont classées dans la catégorie 2).



(a) Règles de décisions

(b) Partitionnement engendré

FIGURE 2.10 – Exemple d'arbre de décision

La variable choisie est celle permettant de réduire au maximum l'hétérogénéité des sous-ensembles obtenus. La variante la plus utilisée pour mesurer l'hétérogénéité des nœuds est basée sur l'entropie de Shannon (Quinlan, 2014) mais on peut aussi citer l'indice de diversité de Gini. Lors de la phase de test, les données sont classées en fonction de la classe majoritaire du sous-espace dont elles font partie. Les arbres de décision ont été longuement étudiés durant cette thèse et seront expliqués en détail lors du chapitre 4.

Forêts aléatoires Les arbres de décision étant très sensibles aux données d'apprentissage, il est naturel de vouloir les combiner dans le cadre de méthodes ensemblistes afin de gagner en robustesse. C'est ce que font les *forêts aléatoires* en combinant des dizaines voire des centaines d'arbres. Pour chacun des arbres, l'ordre d'utilisation des variables est choisi aléatoirement.

Lors de la phase de test, la classe prédite par la majorité des arbres est choisie. L'utilisation d'un grand nombre d'arbres permet de faire des analyses statistiques sur l'importance de chaque caractéristique (plus d'informations au chapitre 4).

Développée il y a vingt ans par Breiman (2001), cette méthode est vite devenue une référence pour la classification supervisée. La méta-analyse menée par Fernández-Delgado *et al.* (2014) étudiant 179 variantes de méthodes de classification supervisée sur 121 jeux de données a ainsi montré l'efficacité des forêts aléatoires. Plus d'informations sur l'utilisation de cette méthode sont disponibles dans l'article de Biau et Scornet (2016).

2.1.2 Les méthodes non supervisées

L'apprentissage non supervisé ne suppose aucune appartenance *a priori* des données à une classe. Il s'agit donc d'extraire des connaissances sans hypothèses prédéfinies sur la structure de ces données et de mettre en évidence des groupes homogènes, ou des règles d'association dans une base de données.

Classification non supervisée (*clustering*)

En apprentissage non supervisé, les données ne sont pas classées et les méthodes de clustering ne consistent donc pas à prédire le groupe d'appartenance d'une donnée, mais au contraire à identifier et créer ces groupes.

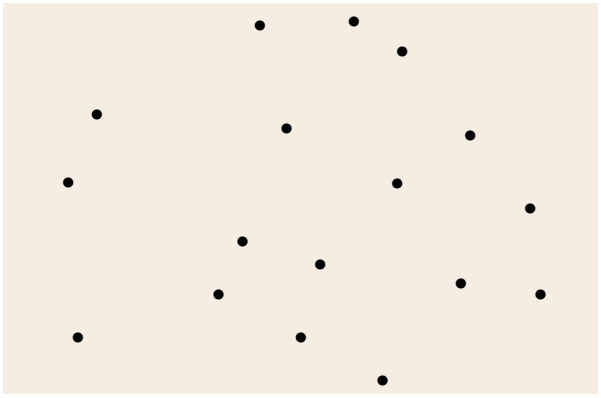
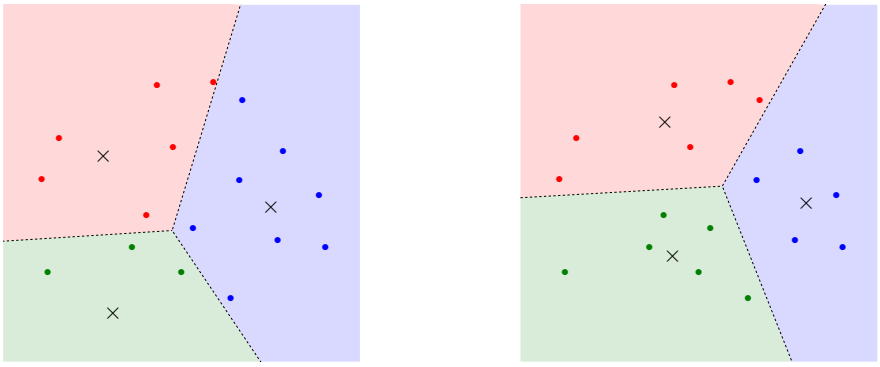


FIGURE 2.11 – Comment classer ces données ?

K-moyennes L'algorithme le plus connu pour partitionner des données est appelé *k-moyennes*. Après avoir défini k, le nombre de classes à obtenir, k points sont créés aléatoirement dans l'espace des données, représentant les centres initiaux des groupes. Chaque donnée est ensuite associée au centre le plus proche, ceux-ci étant ensuite remplacés par les moyennes des groupes ainsi obtenus. Les données sont ensuite associées aux nouveaux centres les plus proches puis l'opération est répétée jusqu'à convergence.

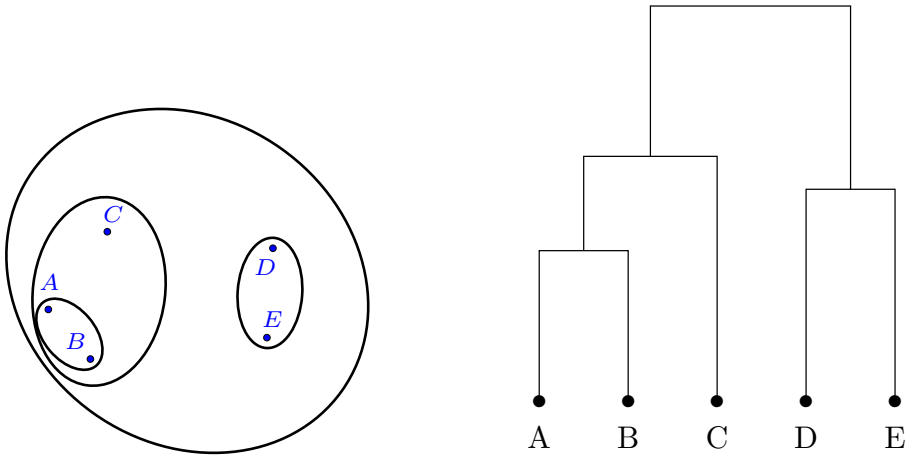


(a) Situation initiale (b) Situation finale

FIGURE 2.12 – Exemple de classification

Cette méthode présente l'avantage d'être rapide à implémenter et à exécuter. Cependant, l'algorithme converge vers un minimum local et il n'y a donc aucune garantie sur l'optimalité de la classification obtenue. Des critères de qualité permettent néanmoins de comparer différents clustering, sur la base de concepts de compacité et de séparation : une méthode sera performante si, géométriquement, les groupes obtenus sont homogènes (inertie intra-classe) et bien séparés (inertie inter-classe).

Classification hiérarchique Une méthode alternative, mais plus longue à exécuter, est la classification hiérarchique, qui est très majoritairement utilisée dans sa version ascendante. Chaque donnée va initialement être considérée comme étant un groupe, formant un singleton. Les deux groupes les plus proches vont ensuite être fusionnés, et leur distance va correspondre au niveau du regroupement dans l'arbre hiérarchique, appelé dendrogramme. Cette étape va ensuite être répétée jusqu'à obtenir un seul groupe contenant toutes les données.



(a) Visualisations des groupes (b) Arbre de classification

FIGURE 2.13 – Exemple de classification hiérarchique

Plus d'informations sur l'exécution et l'utilisation de cette méthode sont disponibles au chapitre 5.

2.1.3 Les méthodes par renforcement

Dans les méthodes de renforcement, nous cherchons à améliorer le comportement (appelé *politique*) d'un agent dans un environnement. Soit S l'ensemble des états possibles de l'agent et A l'ensemble des actions possibles. Le but des méthodes à renforcement est d'optimiser la fonction de récompense $Q : A \times S \mapsto \mathbb{R}$.

Il existe deux catégories de méthodes à renforcement :

1. Les méthodes *Off-policy* qui mettent à jour les récompenses à chaque itération. La plus connue de ces méthodes est le *Q-learning* (Watkins et Dayan, 1992). Une fois la convergence atteinte, l'action choisie, pour un état s donné, est celle apportant la plus grande récompense.

2. Les méthodes *On-policy*, fixent la fonction de récompense. Cette fonction est ensuite testée pendant une série d'itérations, appelée *simulation*, puis, mise à jour. La méthode SARSA (*State Action Rewards State Action*), initiée par Rummery et Niranjan (1994) est la plus connue. C'est également une méthode du type *Temporal Difference Learning* (TD-Learning), où la fonction de récompense obtenue à l'étape $k+1$ est dépendante de la fonction de récompense obtenue à l'étape k, ce qui accélère généralement la convergence de l'apprentissage.

Un résumé sur les fonctions de récompenses utilisées, les équations qui en découlent et les variantes des méthodes d'apprentissage par renforcement est disponible dans l'article de Gosavi (2009). Un livre beaucoup plus complet et récent sur les problèmes de renforcement a été rédigé par Sutton et Barto (2018).

2.1.4 Aller plus loin

Les méthodes d'apprentissage statistique sont multiples et possèdent chacune de nombreuses variantes. Cette section ne représente donc qu'un échantillon des techniques utilisées. Les livres de Bishop (2006) et de Murphy (2012) constituent un excellent moyen de compléter ses connaissances sur le sujet.

2.2 Union de deux disciplines complémentaires

Sortir de son champ de compétence, et s'inspirer de travaux d'autres domaines permet d'imaginer de nouvelles méthodes à la croisée des disciplines. Dans cette section, nous allons voir les travaux utilisant les connaissances en apprentissage statistique pour améliorer la résolution de problèmes d'optimisation.

2.2.1 Une frontière floue

Les domaines de la recherche opérationnelle et de l'apprentissage automatique semblent distincts. Cependant, en y regardant plus en détail, on peut remarquer qu'il existe de l'apprentissage automatique dans certaines méthodes d'optimisation, et beaucoup d'optimisation dans les méthodes d'apprentissage automatique. En effet, les méthodes d'apprentissage supervisé ont pour objectif de minimiser une fonction d'erreur, il s'agit donc entièrement d'un problème d'optimisation. Les réseaux de neurones utilisent la descente

de gradient, jusqu'à convergence, il s'agit donc d'une méthode à recherche locale. Les méthodes ALNS adaptent l'utilisation de leurs voisinages en fonction de leurs performances. Elles utilisent donc de l'apprentissage pour s'améliorer. Enfin, les méthodes basées sur les populations (algorithme génétique, colonie de fourmis, etc.) s'améliorent en se basant sur la mémoire des bonnes solutions, pour en construire de nouvelles. Ces algorithmes sont donc proches des méthodes d'apprentissage automatique. En particulier, la méthode des colonies de fourmis modifie le poids des arêtes en fonction des valeurs des solutions obtenues avec. Il s'agit donc d'une méthode d'optimisation basée sur l'apprentissage par renforcement.

Il y a donc régulièrement une inspiration mutuelle entre les domaines de l'apprentissage et de l'optimisation.

2.2.2 Optimiser l'apprentissage

Avant d'étudier comment l'apprentissage automatique peut aider la recherche opérationnelle, il est intéressant de voir qu'il existe des études faisant le raisonnement inverse.

Par exemple, nous avons vu précédemment (Figure 2.4(b)) qu'il est parfois pertinent d'ajouter une dimension pour améliorer la performance d'une méthode supervisée. Cependant, des dimensions inutiles augmentent le temps d'exécution des algorithmes et peuvent perturber l'apprentissage de ces méthodes. De plus, réduire les dimensions permet de réduire la quantité d'informations à interpréter, et donc d'améliorer la compréhension des données. Il est donc important de choisir les bonnes dimensions à étudier. Dans son article, Mousin *et al.* (2016) propose une méthode à base de recherche tabou pour optimiser le choix des dimensions à conserver.

Il existe de nombreux livres et synthèses expliquant comment utiliser la recherche opérationnelle dans le but d'améliorer l'apprentissage statistique (Sra *et al.*, 2012; Corne *et al.*, 2012; Talbi, 2016; Bottou *et al.*, 2018), montrant ainsi l'importante collaboration existant entre ces deux domaines de recherches.

2.2.3 Apprentissage pour la recherche opérationnelle

Il existe deux manières distinctes d'utiliser l'apprentissage automatique pour la résolution d'une instance. La première consiste à étudier en direct les données récoltées au début de l'utilisation d'un solveur, puis utiliser ses informations pour améliorer la fin de la résolution, on parle alors d'*apprentissage en ligne*. Il est également possible d'utiliser

les informations obtenues sur des instances précédemment résolues pour résoudre plus efficacement les instances futures, il s'agit alors d'*apprentissage automatique*.

Apprentissage pour des utilisations spécifiques

Optimisation multi-objectif L'optimisation multi-objectif est un champ de plus en plus étudié en recherche opérationnelle dont le but est d'obtenir tous les compromis existant entre plusieurs fonctions objectifs. Plus le nombre de fonctions objectifs est important, plus la valeur de chaque solution est longue à calculer, et plus il y a de compromis à trouver. Il est donc important de n'étudier que les fonctions objectifs nécessaires, afin de gagner et clarté et réduire le temps d'exécution des différentes méthodes de résolution.

Dans leurs travaux, Saxena *et al.* (2012) ont essayé de réduire le nombre de fonctions objectifs en explorant la structure des données d'apprentissage via des analyses factorielles. Ces études ont permis de ne garder que les objectifs nécessaires, et donc de réduire significativement le temps de résolution des instances testées.

Une autre approche, menée par Breaban et Iftene (2015) consiste à utiliser des méthodes statistiques pendant la résolution de l'instance, on parle alors d'*apprentissage en ligne*. Pour cela, les solutions obtenues ont été partitionnées, et seules les fonctions objectifs les plus représentatives des compromis sont conservées, pour chaque partition.

Apprentissage pour les méthodes de séparation et évaluation Les méthodes de séparation et évaluation sont très utilisées pour résoudre de manière exacte des problèmes d'optimisation. Le temps d'exécution de ces algorithmes étant fortement impacté par l'ordre d'étude des sous-ensembles et la manière dont ils sont divisés, il est nécessaire de bien choisir sa politique de branchement.

Beaucoup de méthodes se basent sur l'apprentissage du choix de la politique à utiliser sur un groupe d'instances, puis testent l'efficacité sur un autre groupe d'instances, on parle alors d'*apprentissage hors ligne*.

Plusieurs méthodes d'apprentissage supervisé ont été testées, comme les arbres aléatoires (Alvarez *et al.*, 2014), la régression logistique (Khalil *et al.*, 2017b), permettant de prédire l'ordre optimal d'étude des nœuds, ou encore le SVM (Khalil *et al.*, 2016) pour choisir quelle variable étudier pour chaque nœud. Les travaux de Marcos Alvarez *et al.* (2016) utilisent une régression linéaire, basée sur les informations récoltées au début de l'exécution de l'arbre pour guider la suite de la résolution. Une synthèse des techniques d'apprentissage statistique pour les méthodes à branchements est disponible dans l'article

de Lodi et Zarpellon (2017).

Apprentissage hors ligne

Dans l'apprentissage hors ligne, le but est de déduire des règles à partir d'un ensemble d'instances, et de les généraliser pour les instances à venir. Cette manière de procéder présente l'avantage d'être effectuée avant de connaître les nouveaux problèmes à résoudre, le temps d'apprentissage est donc séparé du temps d'exécution par le solveur.

Analyses statistiques Le premier intérêt des sciences des données est de mieux comprendre les informations dont nous disposons. Avant d'utiliser des méthodes avancées d'apprentissage statistique, il est intéressant de voir que les méthodes simples, comme la visualisation de plusieurs solutions intéressantes, permet parfois une plus grande avancée.

Ainsi, Mousin *et al.* (2017) ont remarqué que sur des problèmes d'ordonnancement, il n'est pas rare qu'une tâche i soit toujours exécutée juste avant une tâche j pour chacune des solutions de bonne qualité étudié. Les auteurs ont donc mis au point une méthode fusionnant les deux tâches (la tâche résultante est appelée *super-job*) et réduisant ainsi la taille du problème à résoudre. Au fur et à mesure des recherches locales, les super-jobs sont détruits, pour une meilleure diversité des solutions, et d'autres sont créés en observant les nouvelles solutions.

En étudiant la topologie des bonnes solutions des problèmes de coloration de graphe, Porumbel *et al.* (2010) ont remarqué que celles-ci sont souvent regroupées par îlots dans plusieurs zones. Ils ont ainsi développé deux méthodes à voisinages, la première diversifiant les solutions pour augmenter le nombre d'îlots rencontrés, et la deuxième intensifiant la recherche, pour les explorer en détail.

Une autre approche, testée par Schiavinotto et Stützle (2007) est de calculer plusieurs métriques calculant la distance entre deux permutations. En particulier, certaines distances correspondent au nombre de modifications d'un type particulier nécessaire pour changer la première permutation pour que celle-ci devienne la deuxième permutation. Les solutions du TSP étant souvent modélisées en permutation, cette analyse a permis de constater la similitude entre certains voisinages.

Les minima locaux sont le problème principal des méthodes à recherche locale. Dans leurs travaux, Ochoa et Veerapen (2016, 2018) étudient la structure des minima locaux afin de chercher des méthodes permettant d'en sortir.

Enfin, l'article de Solano-Charris *et al.* (2015) met en valeur l'intérêt de tests statistiques pour choisir la meilleure méthode à utiliser pour résoudre certains problèmes. En testant plusieurs méta-heuristiques sur des problèmes de VRP, leur analyse statistique a permis de mettre en évidence la robustesse des méthodes à recherche locale basée sur une population de solution.

Réglage des solveurs Nous avons déjà évoqué plusieurs études du bon choix d'heuristiques ou de paramétrage de méthodes d'optimisation. Il s'agit d'un problème d'optimisation appelé *problème de sélection d'algorithme*, imaginé par Rice (1976).

Contrairement aux problèmes classiques d'optimisation où la fonction objectif est généralement rapide à calculer, ici, pour connaître les performances d'un algorithme, il est nécessaire de l'exécuter, ce qui risque est particulièrement long. L'apprentissage automatique est donc utilisé pour prédire l'algorithme à utiliser (Kotthoff *et al.*, 2012).

Afin de faciliter cette recherche d'algorithme, un outil nommé LLAMA a été développé en R et utilise de nombreuses bibliothèques d'analyses statistiques pour choisir le bon jeu de paramètre (Kotthoff, 2013).

Dans le cadre de la recherche de la meilleure méta-heuristique pour une instance donnée, les recherches récentes concernant les variantes du VRP sont principalement focalisée sur les réseaux de neurones (Tyasnurita *et al.*, 2017; Gutierrez-Rodríguez *et al.*, 2019).

Certaines caractéristiques du TSP ont été étudiées par Kanda *et al.* (2016) pour voir si des méthodes d'apprentissage supervisé (arbre de décision, k plus proches voisins et perceptron multicouche) permettent de déterminer la méta-heuristique la plus efficace pour une instance donnée.

Cette problématique est également présente lors de la résolution de problèmes sous forme d'un programme linéaire, que ce soit pour choisir le paramétrage de Cplex (Bonami *et al.*, 2018), ou s'il faut utiliser une méthode de décomposition (Kruber *et al.*, 2017).

Plus d'informations sur ces méthodes sont disponibles dans le papier de Kotthoff (2016).

Apprendre grâce à un historique de solutions L'intérêt principal de l'apprentissage hors ligne, est le fait de pouvoir exécuter les algorithmes d'apprentissage bien avant de recevoir l'instance à résoudre. Le temps d'exécution de ces méthodes n'est donc pas un problème, ce qui permet l'utilisation de jeux de données important, basé sur des solutions d'anciennes instances déjà résolues. Nous verrons ici quelques techniques utilisant ces

bases d'entraînement.

Avec le développement des méthodes à réseaux de neurones, nombreux sont ceux à avoir essayé de les utiliser pour apprendre comment obtenir la solution optimale d'un problème combinatoire. Le TSP est particulièrement étudié, par exemple par Bello *et al.* (2016) qui combinent réseau de neurones et apprentissage par renforcement pour générer des solutions de bonne qualité. Cependant, l'apprentissage sur des instances d'une centaine de clients est extrêmement long, ce qui rend difficile le passage à l'échelle (Joshi *et al.*, 2019; da Costa *et al.*, 2020).

En utilisant des règles d'associations pour des problèmes d'ordonnancement (*si le job* 2 *est avant le job* 4*, alors le job* 3 *doit être après le job* 1, etc.), Nasiri *et al.* (2019) génèrent un ensemble de solutions servant ensuite de points de départ pour des méthodes à base de population.

À cause de leur structure particulière, il est difficile d'évaluer avec précision la similarité entre deux graphes, et donc par extension, les liens entre deux instances représentées sous cette forme. En utilisant les méthodes de *graph embeded*, qui transforment les graphes en vecteurs, Khalil *et al.* (2017a) ont réussi à contourner ce problème pour apprendre sur les k instances les plus proches les règles permettant de construire une bonne solution sur plusieurs problèmes de graphes, dont le TSP.

Il est fréquent que les solutions retournées par les solveurs ne correspondent pas aux attentes des opérateurs, dû à des contraintes techniques difficiles à modéliser. En apprenant sur une base de solutions modifiées par les opérateurs pour des problèmes de VRP, Canoy et Guns (2019) ont réussi, via des modèles de Markov, à pondérer les arêtes en fonction de leur utilisation, guidant ainsi la recherche vers des solutions qui correspondent plus aux attentes demandées par les clients.

Enfin, Arnold et Sörensen (2019b) ont généré et étudié différentes caractéristiques issues de solution de CVRP, ainsi que de leurs instances associées. En utilisant des méthodes d'apprentissage automatique classifiant les solutions en non-optimales ou quasi-optimales, les auteurs ont réussi à identifier deux caractéristiques à minimiser pour obtenir des solutions de bonne qualité : le nombre de croisements entre les tournées et la largeur des tournées. Ces connaissances ont ensuite permis de développer une nouvelle méthode de recherche locale (Arnold et Sörensen, 2019a), pénalisant les arêtes se croisant, ainsi que les tournées dont la largeur est trop importante.

Apprentissage en ligne

Dans cette section, nous verrons les méthodes utilisant les connaissances extraites pendant le résolution du problème courant afin d'adapter le solveur aux spécificité de l'instance à traiter.

Reactive Search Nous avons vu dans la section précédente qu'il est souvent difficile de choisir le paramétrage donnant les meilleures performances. Les méthodes *reactive tabu search*, ou RTS (Battiti et Tecchiolli, 1994) et *reactive GRASP* (Prais et Ribeiro, 2000) modifient le paramétrage pendant la résolution. La méthode RTS adapte la taille de liste tabou pour améliorer la recherche : tant que la recherche locale mène à des solutions intéressantes, le nombre de solutions tabou est faible. Cependant, dès que la méthode est bloquée dans un minimum local, le nombre de solutions tabou augmente pour forcer à changer d'aire de recherche. Une fois le minimum local quitté, le nombre de solutions tabou est réduit progressivement.

Concernant la méthode reactive GRASP, c'est le paramètre α, correspondant à la proportion d'aléatoire dans les solutions créées, qui évolue, pour garantir le bon rapport équilibre entre la diversité des solutions, et leur qualité.

Extraction de structures Une idée intuitive est de supposer que les solutions de bonne qualité ont des caractéristiques en commun. Certains algorithmes ont été conçu avec cette idée : ils exécutent une première fois une heuristique, puis essayent d'extraire des caractéristiques communes aux meilleures solutions pour guider la création de nouvelles solutions.

Par exemple, Tarantilis et Kiranoudis (2002) ont créé la méthode du *bone route* : après avoir généré et amélioré plusieurs solutions, l'algorithme va extraire les séquences les plus fréquentes dans les solutions de bonne qualité. Une partie de ces séquences sera imposée dans la création de nouvelles solutions. Cette idée va également être utilisée par Ribeiro *et al.* (2006) pour résoudre le *set packing problem*, ou par Arnold *et al.* (2019b) qui vont augmenter de manière itérative la taille des séquences à conserver.

Apprentissage par renforcement La méthode de référence pour l'apprentissage en ligne est l'apprentissage par renforcement, qui évolue à chaque itération. Un exemple d'approche par renforcement est utilisé par Silva *et al.* (2019) pour des problèmes de VRPTW et d'ordonnancement, via l'utilisation de recherches à voisinage multiples. Leur matrice état×actions correspond au voisinage à utiliser en fonction de celui qui vient d'être exécuté.

Au début des années 2000, les méthodes par renforcement étaient principalement utilisées pour détecter un minimum local, et lisser la fonction objectif (voir figure 2.14) afin de poursuivre la recherche (Boyan et Moore, 2000), ainsi le minimum local n'existe plus et la recherche peut se poursuivre. Cette idée a donné lieu à la méthode de *guided local search*, ou GLS (Voudouris et Tsang, 1999, 2003).

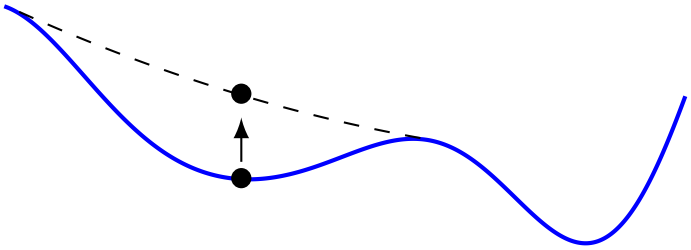


FIGURE 2.14 – Exemple de lissage

Décomposition du problèmes Une autre approche consiste à diviser le problème à résoudre en sous problèmes plus simples. Par exemple, Qi *et al.* (2012) utilise la méthode des k-médoïdes (une variante des k-moyennes où les points centraux font partie des données) pour diviser les clients par zone géographique et par date d'accessibilité dans les problèmes de VRPTW. Les tournées sont ensuite optimisées en parallèle. Leur méthode principale étant un algorithme génétique et les clients servant de centre constituant l'ADN des solutions, un grand nombre de division peut ainsi être testé.

2.3 Conclusion

Au cours des deux dernières décennies, de nombreuses idées ont émergé pour faire collaborer les domaines de l'apprentissage statistique et de la recherche opérationnelle. Cependant, ces méthodes hybrides doivent encore faire leurs preuves. En effet, les méthodes de deep learning, bien que très étudiées, ont encore du mal à passer à l'échelle pour un problème aussi simple que le TSP (Joshi *et al.*, 2019; da Costa *et al.*, 2020). Une méta-analyse rédigée par Turkeš *et al.* (2019) a montré que l'apprentissage par renforcement utilisé dans les méthodes ALNS n'est finalement pas statistiquement plus efficace que l'utilisation aléatoire des voisinages.

De manière générale, les deux angles d'approche des méthodes hybrides comportent des barrières qui n'ont pas été levées :

— Les méthodes d'apprentissage hors ligne dépendent du lien entre les instances et les solutions. Si deux instances similaires ont des solutions optimales différentes, il est difficile d'y énoncer des règles générales.

— Les méthodes d'apprentissage en ligne sont influencées par les premières solutions obtenues. Ainsi, les méthodes en ligne risquent d'être biaisées par les premiers minima locaux obtenus, empêchant toute diversification des solutions.

Bien que prometteuses, les méthodes hybrides doivent encore être étudiées pour pouvoir rivaliser avec des méthodes issues de décennies de recherche.

Une synthèse des méthodes hybrides a été rédigée par Bengio *et al.* (2018) et permet de mieux comprendre les enjeux qui y sont associés.

Chapitre 3

Un solveur stable et adaptatif pour des variantes du CVRP

Très tôt dans la thèse, il a été décidé de créer un solveur robuste et rapide. En d'autres termes, ce solveur doit être capable de trouver en quelques minutes des solutions de bonne qualité sur des instances de topologies différentes, réelles et de grande taille.

Dans ce chapitre, nous détaillerons toutes les particularités de ce solveur, intitulé RADOS (*Routing And Delivery Optimisation Solver*). Celui-ci étant basé sur des recherches locales multiples, nous commencerons par détailler les étapes de création des solutions initiales, avant de décrire l'ensemble des voisinages utilisés. Une fois les principales composantes du solveur décrites, nous détaillerons comment celui-ci est agencé. Enfin, nous terminerons par des tests sur deux jeux d'instances difficiles.

Sommaire

3.1 Création de la solution initiale

Le solveur se base principalement sur l'amélioration d'une solution créée de manière itérative. Dans cette section, nous détaillerons l'algorithme utilisé pour générer les solutions initiales.

3.1.1 L'algorithme de Clarke and Wright

L'algorithme le plus utilisé pour créer rapidement une solution de qualité correcte est l'algorithme de Clarke and Wright. Celui-ci commence par assigner une tournée par client (figure 3.1). Ensuite, l'algorithme va essayer de fusionner deux tournées afin de réduire le coût total de transport. L'astuce consiste à remarquer (voir figure 3.2) que le gain obtenu lors de la fusion ne dépend que du dernier client de la tournée 1, et du premier client de la tournée 2.

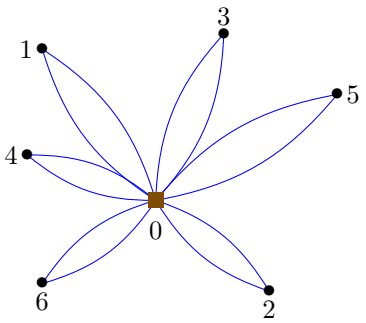


FIGURE 3.1 – Situation initiale de l'algorithme de Clarke and Wright

Notons $d_{i,j}$ le coût pour parcourir l'arête (i, j). Par convention, le dépôt est le nœud 0. Le gain, appelé *saving*, lors de la fusion entre deux tournées s'obtient avec le calcul suivant : $s_{i,j} = d_{i,0} + d_{0,j} - d_{i,j}$, où i représente le dernier client de la première tournée, et j le premier client de la deuxième tournée.

Dans l'exemple en figure 3.2, le gain lors de la fusion des tournées bleues et rouges est : $s_{1,3} = d_{1,0} + d_{0,3} - d_{1,3}$.

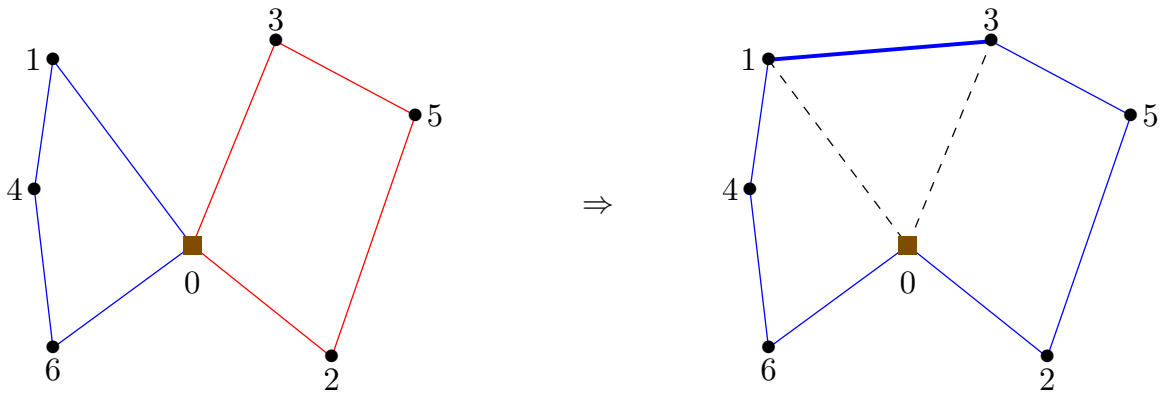


FIGURE 3.2 – Fusion des tournées 6-4-1 et 3-5-2

L'algorithme applique à chaque étape la fusion avec le meilleur *saving*, sous réserve de respecter la contrainte de capacité. L'algorithme s'arrête lorsqu'il ne reste aucun *saving* de valeur positive. Cette situation apparaît au moment où aucune fusion permettant de réduire les coûts n'est réalisable.

3.1.2 Réparation d'une solution

Il est fréquent que la solution retournée par l'algorithme de Clarke and Wright ne soit pas réalisable. En effet, cet algorithme ne garantit aucunement de respecter la limitation K du nombre de véhicules disponibles. C'est particulièrement le cas pour les instances où la somme totale de la demande des clients est proche de la capacité totale des véhicules.

Il est donc nécessaire d'éliminer les tournées excédentaires, et d'affecter leurs anciens clients associés aux tournées restantes. Afin d'augmenter les chances de réinsertion, on calcule pour chaque tournée la plus forte demande de ses clients.

Par exemple, imaginons une instance où uniquement $K = 3$ véhicules sont disponibles, avec une capacité de 15, mais où l'algorithme de Clarke and Wright a fourni une solution avec 4 véhicules. Les demandes respectives des clients servis sont :

4, **5**, 1	**3**, 2, 2, **3**	**2**, 1, **2**, **2**, 1	3, 1, **4**, 2, 1

Les valeurs maximales des demandes associées à chaque tournée sont donc respectivement 5, 3, 2, 4.

La priorité est de conserver les tournées avec les clients à forte demande, car ceux-ci sont difficiles à insérer dans une tournée. Les K tournées avec la plus forte demande maximale sont conservées. Les autres sont détruites, et les clients sont réaffectés dans les différentes tournées, en commençant par les clients à forte demande. Dans le cas où un client ne peut être inséré dans aucune tournée, on s'autorise à enlever un client dont la demande est inférieure.

Exemple : On cherche à insérer un client dont la demande est 5, dans une tournée de capacité 15, dont la demande totale est 14. Il est donc impossible d'insérer ce client. Cependant, la tournée contient un client dont la demande est 4. Ce client est remplacé par celui dont la demande est 5. Ainsi, il reste toujours un client non affecté, mais avec une demande plus faible, donc plus facile à insérer.

3.2 La recherche locale : une composante essentielle

Le solveur utilisé est principalement basé sur la recherche locale par voisinages multiples, associés à une recherche tabou, noté **MNS-TS** (Multiple Neighborhood Search - Tabu Search). Ce nouveau solveur, appelé **RADOS** (*Routing And Delivery Optimisation Solver*) est inspiré de la méthode utilisée par Soto *et al.* (2017) pour les problèmes de tournées ouvertes à dépôts multiples. Les auteurs ont en effet créé une méthode permettant de générer la plupart des voisinages connus avec une représentation du type *path Move* (plus de détails à la page 70), facilitant ainsi leur utilisation. Nous avons donc utilisé cette idée pour générer et tester un ensemble de voisinages, avant de les réimplémenter, en ignorant la notion de path Move, réduisant ainsi la quantité de calculs associés à chaque mouvement.

Cette section décrit en détail chacun des voisinages, les astuces permettant d'augmenter leur efficacité, ainsi que la façon dont elles sont combinées à une recherche tabou.

3.2.1 Recherche locale : le principe

La méthode la plus utilisée pour obtenir en un minimum de temps des solutions de bonne qualité est la descente de gradient. Cette méthode consiste à étudier la pente autour d'une solution, puis se diriger dans la direction de plus grande descente.

L'équivalent en optimisation combinatoire consiste à étudier les solutions accessibles après une légère perturbation d'une solution choisie. La perturbation utilisée est appelée

mouvement, les solutions obtenues sont appelées *voisins* et l'ensemble des solutions admissibles est appelé *voisinage*. Le gain de qualité des solutions voisines est donc équivalant à la pente de la descente. La solution de meilleure qualité correspond donc à celle de la plus grande pente, et est la nouvelle solution de référence. Ce procédé est répété jusqu'à absence d'amélioration (voir figure 3.3). La solution ainsi obtenue est appelée minimum local.

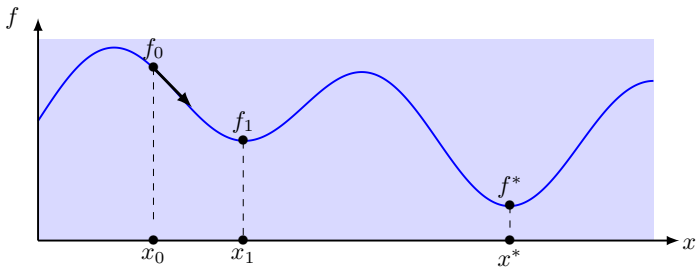


FIGURE 3.3 – Exemple de descente par recherche locale

Sur la figure 3.3, la solution x_0 a un coût de f_0. La flèche représente la direction réduisant au maximum le coût de la solution. Après plusieurs étapes de descente, on obtient la solution x_1, qui est un minimum local : il est impossible d'améliorer x_1 en lui appliquant un unique mouvement. Cette solution n'est pas optimale, car moins intéressante que la solution x^*.

3.2.2 De nombreux voisinages

Pour éviter de s'arrêter au premier minimum local rencontré, de nombreux voisinages ont été ajoutés. En effet, la notion de minimum local est dépendante de celle de voisinage. Changer de voisinage permet de modifier le minimum local associé à la solution courante. Disposer de plusieurs voisinages est donc un excellent moyen d'améliorer la qualité des solutions obtenues. En contrepartie, chaque voisinage supplémentaire augmente le temps d'exécution du solveur.

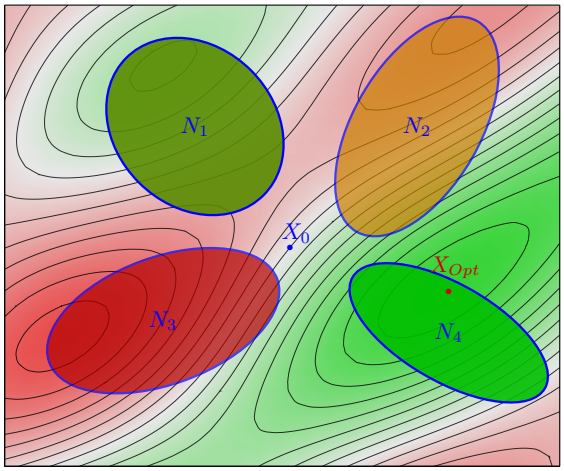


FIGURE 3.4 – Représentation des différents voisinages

Exemple Soit un problème d'optimisation dont la figure 3.4, donne une représentation de l'espace des solutions. Les solutions en vert sont des solutions de bonne qualité, les solutions en rouge sont de mauvaise qualité et les solutions en blanc sont de qualité intermédiaire. Soit X_0 un point généré aléatoirement, et supposons que l'on dispose de 4 mouvements (que l'on notera M_1, M_2, M_3 et M_4) permettant de modifier la solution X_0. Chacune des solutions admissibles après application de ces mouvements forme les ensembles notés respectivement N_1, N_2, N_3 et N_4. Le mouvement M_3 permet d'explorer la zone N_3, ne contenant aucune solution de coût inférieur à X_0. La solution X_0 est donc un minimum local, par rapport au mouvement M_3. Le mouvement M_1 permet d'obtenir des solutions de meilleure qualité. Cependant, la plupart des solutions proches de la zone N_1 sont de moins bonne qualité. Utiliser ce voisinage risque de nous bloquer dans un minimum local. Le mouvement M_4 permet d'explorer la zone N_4, contenant la solution optimale X_{opt}. Le mouvement M_2 permet d'obtenir une solution de meilleure qualité, non optimale. Il sera donc nécessaire d'explorer un nouveau voisinage après application du mouvement M_2.

Le VRP est parfois considéré comme la combinaison d'un problème de TSP (livraison la moins coûteuse d'un ensemble de clients) et d'un problème d'affectation (affectation la moins coûteuse des clients à un ensemble donné de véhicules). Cette vision se reflète dans les types de voisinages utilisés. En effet, ce solveur utilise principalement deux types de voisinages :

— Les voisinages *intra-route*, se concentrant sur la partie livraison.

— Les voisinages *inter-route*, se concentrant sur l'affectation clients/véhicules.

Voisinages intra-route

Ces voisinages représentent les modifications n'impactant qu'une tournée à la fois. Ils modifient donc uniquement l'ordre dans lequel les clients sont servis au sein d'une tournée. Ils sont très utiles pour réduire le coût de chaque tournée.

Trois types de mouvements intra-route ont été implémentés :

— Relocate

— Swap

— 2-opt

Relocate (intra-route) Le mouvement relocate change la position d'un client au sein de la séquence du véhicule associé à ce client.

Par exemple, imaginons la séquence suivante : $0 \rightarrow 4 \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow \mathbf{5} \rightarrow 6 \rightarrow 0$, représentée visuellement par la figure 3.5. Dans cette configuration, le livreur fait un détour important pour livrer le client 5. Pour régler ce problème, il est décalé dans la séquence de livraison, pour être finalement livré entre le client 4 et le client 1. On obtient ainsi la nouvelle séquence : $0 \rightarrow 4 \rightarrow \mathbf{5} \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow 6 \rightarrow 0$.

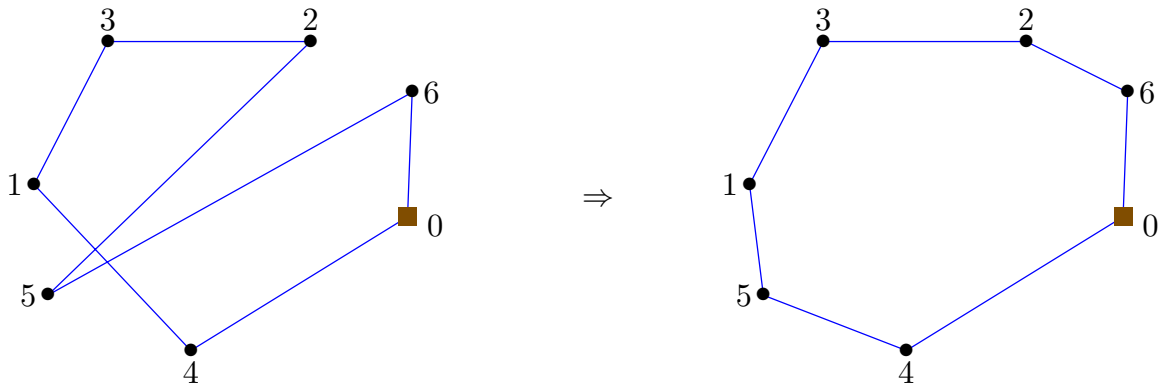


FIGURE 3.5 – Exemple de tournée nécessitant un mouvement Relocate

Soit n_k le nombre de clients de la tournée k, que l'on essaye d'améliorer. Chacun des n_k clients peut être déplacé de sa position vers n'importe laquelle des n_k - 1 positions

restantes. Il y a donc $O(n_k^2)$ possibilité d'amélioration de la tournée k. Trouver un mouvement de réallocation intra-route améliorant est donc de complexité $O(n^2)$. Ce pire cas n'arrive que lorsqu'un seul véhicule est utilisé.

Swap-Exchange (intra-route) Ce mouvement échange la position de deux clients dans la séquence de livraison.

Par exemple, dans la séquence $0 \to 1 \to \mathbf{6} \to 3 \to 4 \to 5 \to \mathbf{2} \to 0$ (voir figure 3.6), échanger les positions des nœuds 2 et 6 change drastiquement le coût de la tournée. La séquence devient donc : $0 \to 1 \to \mathbf{2} \to 3 \to 4 \to 5 \to \mathbf{6} \to 0$ De manière similaire à la réallocation intra-route, trouver un mouvement de swap intra-route est en $O(n^2)$.

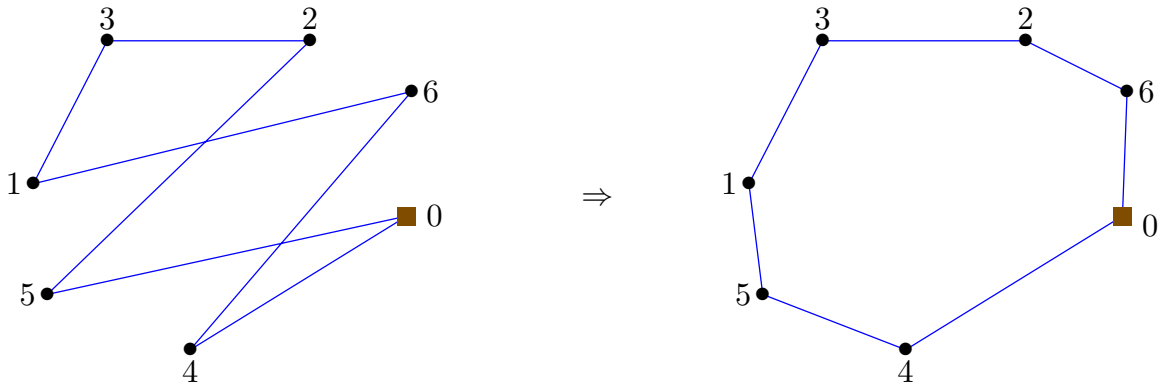


FIGURE 3.6 – Exemple de tournée nécessitant un mouvement Swap

2-opt Le dernier mouvement fait partie de la famille des *k-opt*. Cette famille de mouvements enlève k arêtes d'une tournée et les remplace par k autres arêtes. Plus le nombre k est élevé, plus le nombre de mouvements possibles est important. Une valeur élevée de k augmente les possibilités de tomber sur un minimum local intéressant, mais au prix d'un temps de calcul conséquent.

Le solveur RADOS contient uniquement le 2-opt ; la version la plus simple et rapide des k-opt. Ce mouvement sert principalement à supprimer des croisements au sein d'une tournée.

Exemple : Dans la séquence, $0 \to 4 \to \mathbf{2} \to \mathbf{3} \to \mathbf{1} \to \mathbf{5} \to 6 \to 0$, les arêtes **4-2** et **5-6** se croisent (voir figure 3.7), ce qui, dans le cas euclidien, est la preuve que la solution n'est pas optimale (voir preuve en annexe, page 175). En les remplaçant par les arêtes **4-5** et

2-6, le croisement est supprimé. Il est a noter que l'ordre dans lequel les clients 2, 3, 1 et 5 sont livrés a été inversé. La séquence devient alors : $0 \rightarrow 4 \rightarrow \mathbf{5} \rightarrow \mathbf{1} \rightarrow \mathbf{3} \rightarrow \mathbf{2} \rightarrow 6 \rightarrow 0$

On remarque que dans le codage en séquences, appliquer un mouvement 2-opt revient à choisir un couple de clients (i, j) et à inverser la séquence de livraison les reliant. On obtient donc $O(n^2)$ combinaisons possibles de 2-opt.

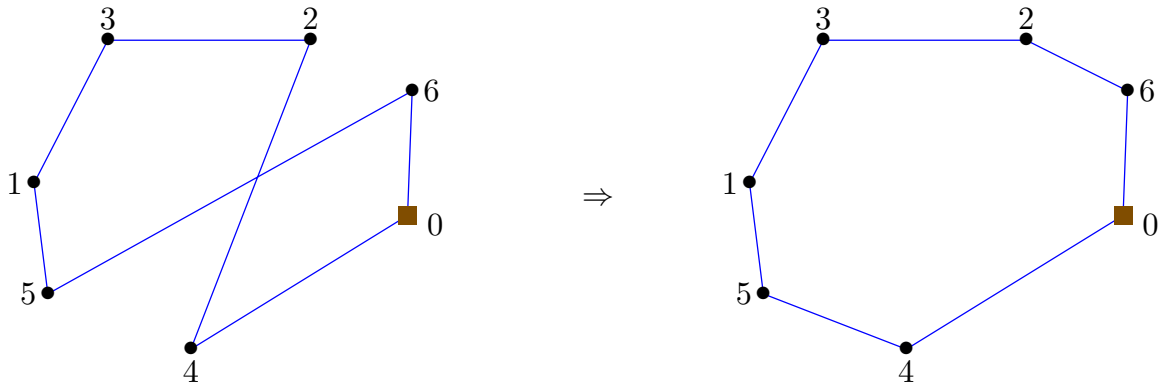


FIGURE 3.7 – Exemple de tournée nécessitant un mouvement 2-opt

Voisinages inter-routes

Les voisinages inter-routes modifient au moins deux tournées. Ils permettent en particulier d'améliorer l'affectation clients/véhicules.

Relocate (inter-route) De manière similaire à sa version intra-route, le relocate inter-route (ou, réallocation inter-route) modifie une solution de manière très localisée. Plus particulièrement, ce mouvement enlève un client d'une tournée, et le place dans une tournée différente, de manière à ce que son insertion soit la moins coûteuse possible.

Par exemple, dans le cas défini par la figure 3.8, les tournées sont représentées par les séquences :

- $0 \rightarrow \underline{\mathbf{8}} \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$
- $0 \rightarrow 6 \rightarrow 7 \rightarrow 9 \rightarrow 10 \rightarrow 0$

La tournée en bleu fait un détour pour livrer le client 8, qui se trouve à proximité des clients 7 et 9, livrés par la tournée rouge. L'affectation clients/véhicules n'est donc pas

optimale. La réallocation inter-route permet d'enlever le client 8 et la tournée bleue, et de l'insérer entre la livraison du client 7 et celle du client 9.

Après application du mouvement, on obtient les deux séquences suivantes :

- $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$
- $0 \rightarrow 6 \rightarrow 7 \rightarrow \underline{8} \rightarrow 9 \rightarrow 10 \rightarrow 0$

L'affectation semble meilleure. Ce mouvement a cependant un inconvénient majeur : lorsque la demande totale des clients est proche de la capacité totale des véhicules, la capacité de chaque tournée est utilisée à son maximum, et il devient difficile d'insérer de nouveaux clients.

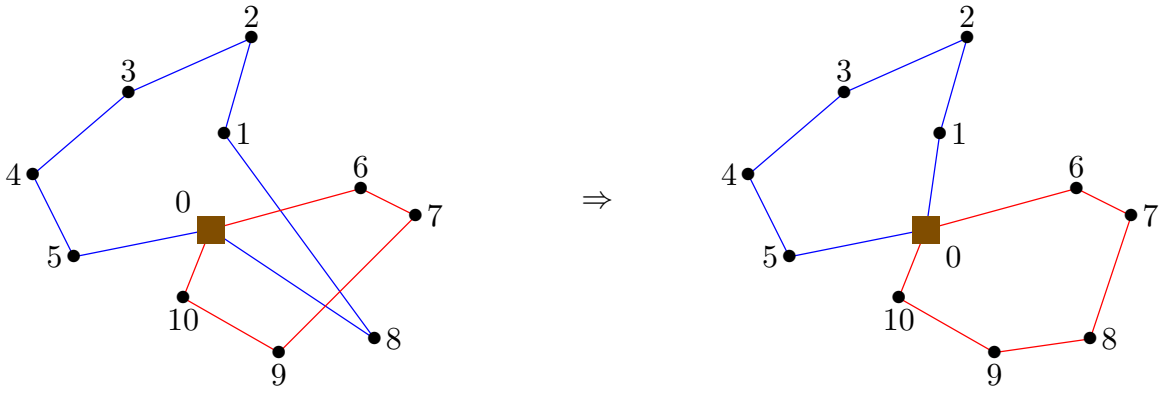


FIGURE 3.8 – Exemple de tournées nécessitant un mouvement relocate

Avec un raisonnement similaire à la réallocation intra-route, on obtient un nombre de combinaisons possibles pour un mouvement de réallocation inter-route de l'ordre de $O(n^2)$.

Swap (inter-route) Tout comme son équivalent intra-route, le swap échange la position de deux clients dans le codage de la solution. Alors que le swap intra-route échange deux clients de la même tournée, le swap inter-route échange deux clients de deux tournées différentes. Ce mouvement est particulièrement utile lorsque la demande est élevée et que les tournées ont une capacité restante insuffisante pour livrer un client supplémentaire. De manière similaire à sa version intra-route, appliquer une recherche locale via ce mouvement est de complexité $O(n^2)$.

Dans l'exemple 3.9, le véhicule de la tournée rouge fait un détour pour livrer le client 1, avant de passer à côté du client 6, qui est finalement livré par le véhicule de la tournée bleue. Échanger les tournées d'affectations des clients 1 et 6 permet donc de réduire la durée de chacune des tournées, et donc la distance totale parcourue. La solution passe donc de cette représentation en séquence :

- $0 \rightarrow \underline{\mathbf{6}} \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$
- $0 \rightarrow \underline{\mathbf{1}} \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 0$

À celle-ci :

- $0 \rightarrow \underline{\mathbf{1}} \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$
- $0 \rightarrow \underline{\mathbf{6}} \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 0$

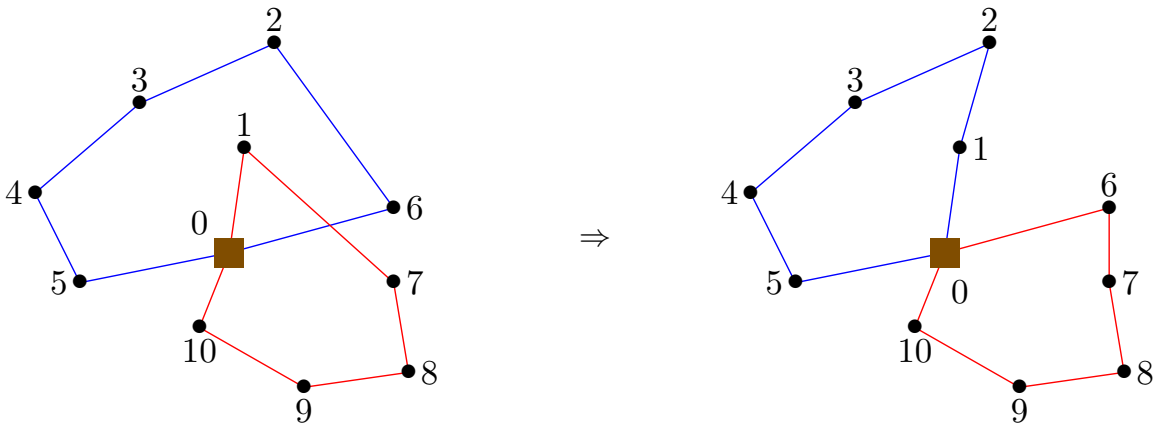


FIGURE 3.9 – Exemple de tournées nécessitant un mouvement Swap

Crossover / 2-opt* Il est fréquent que les heuristiques pour les problèmes de tournées créent des solutions où plusieurs tournées se croisent. Pour régler ce problème, une méthode simple consiste à intervertir les sous-séquences terminant deux tournées. Ce mouvement est appelé crossover. Ce mouvement se fait en supprimant les arêtes (i, j) et (k, l) de croisement entre tournées, et de créer les arêtes (k, l) et (j, l), afin d'accorder à nouveau les 4 sous-séquences. Appliquer un mouvement de crossover revient donc à appliquer un mouvement de 2-opt, mais entre deux tournées différentes, d'où son autre nom : 2-opt*.

La figure 3.10 illustre un cas où le crossover est nécessaire. Les tournées sont représentées par les permutations suivantes :

- $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow \underline{\mathbf{4} \rightarrow \mathbf{5} \rightarrow \mathbf{6}} \rightarrow 0$
- $0 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow \underline{\mathbf{10} \rightarrow \mathbf{11} \rightarrow \mathbf{12}} \rightarrow 0$

En effet, les deux tournées s'entrecroisent au niveau des arêtes (3,4) et (9,10). On applique donc le mouvement de crossover pour supprimer le croisement entre les tournées. En matière de permutation, cela revient à échanger la fin de chacune des séquences :

- $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow \underline{\mathbf{10} \rightarrow \mathbf{11} \rightarrow \mathbf{12}} \rightarrow 0$
- $0 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow \underline{\mathbf{4} \rightarrow \mathbf{5} \rightarrow \mathbf{6}} \rightarrow 0$

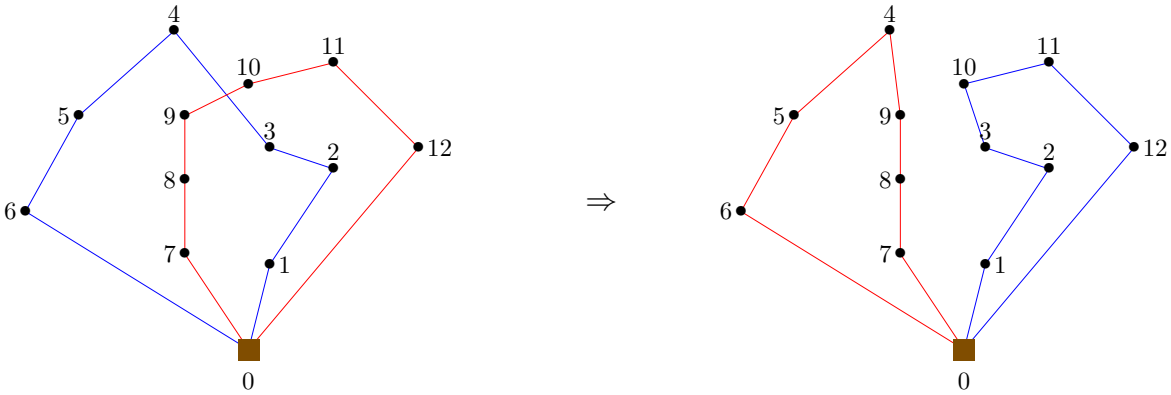


FIGURE 3.10 – Exemple de tournées nécessitant un mouvement de crossover

Path-move Le path-move est un mouvement consistant à déplacer une sous-séquence de clients. Ce déplacement peut s'effectuer au sein d'une tournée, ou entre deux tournées.

Par exemple, dans la figure 3.11, la solution est composée des deux tournées suivantes :

- $0 \rightarrow \underline{\mathbf{8} \rightarrow \mathbf{7}} \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$
- $0 \rightarrow 6 \rightarrow 9 \rightarrow 10 \rightarrow 0$

La sous-séquence $\mathbf{8} \rightarrow \mathbf{7}$ semble causer un allongement important de la tournée bleue. Il semble donc pertinent de déplacer la séquence et de l'insérer dans la tournée rouge, entre la livraison du client 6 et celle du client 9. Les tournées s'effectuant dans un sens de rotation différent, insérer la séquence $\mathbf{8} \rightarrow \mathbf{7}$ ainsi forme un croisement au sein de la tournée rouge, ce qui n'est probablement pas optimal. Pour pallier ce problème, on inverse l'ordre de livraison dans la séquence. On obtient ainsi les séquences :

- $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 0$
- $0 \rightarrow 6 \rightarrow \underline{\mathbf{7} \rightarrow \mathbf{8}} \rightarrow 9 \rightarrow 10 \rightarrow 0$

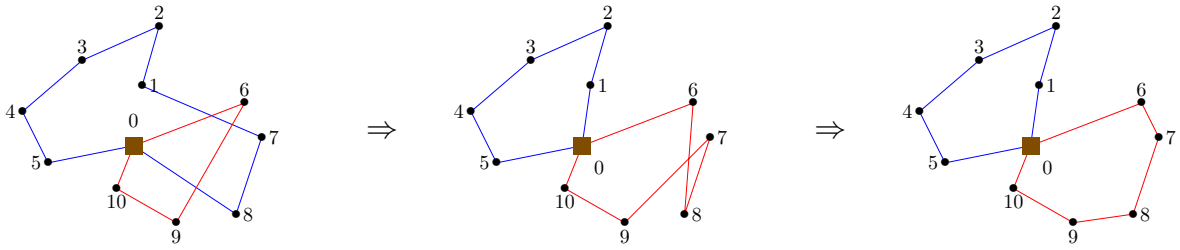


FIGURE 3.11 – Exemple de tournées nécessitant un mouvement pathmove + inversion

Remarque En spécifiant certaines règles sur cette famille de mouvement, on peut retrouver les familles précédentes :

— Un mouvement path-move de taille 1 correspond à un mouvement de réallocation.

— Le mouvement swap peut être généré à partir de deux exécutions de path-move de taille 1.

— Un 2-opt peut être effectué avec un path-move ne changeant pas la position de la séquence, mais effectuant une inversion.

Pour une taille de séquence donnée, il existe $O(n^2)$ mouvements pathmove possibles.

Split La majorité des heuristiques sur les problèmes de tournées sont des méthodes dites "Cluster first, route second". Ces méthodes commencent par trouver une bonne affectation clients/véhicules, puis réduisent le coût de chacune des tournées. Cependant, certaines méthodes procèdent de manière opposée : elles commencent par résoudre le problème en n'utilisant qu'un seul véhicule (figure 3.12). Une fois le tour géant résultant optimisé à la manière d'un TSP, la phase de séparation (appelée SPLIT dans la littérature) peut commencer (figure 3.13).

Les figures 3.12 à 3.14 sont toutes issues de l'article de Prins (2004), qui utilise l'algorithme de séparation pour décoder les solutions dans un algorithme génétique.

Cette méthode reprend la méthode de Bellman-Ford, utilisée pour le plus court chemin. L'idée est la suivante : un nouveau graphe est créé, où chaque arête correspond à une tournée complète entre tous les clients depuis l'arête de départ (non incluse), jusqu'au

client d'arrivée (inclus). Par exemple, l'arête (3,5) dans la figure 3.13 correspond à l'ajout d'une tournée contenant les clients 4 et 5. L'algorithme part du premier nœud (le dépôt), regarde chaque arête partant de ce nœud, et met à jour le coût minimum pour aller au nœud d'arrivée, ainsi que le nœud de départ de l'arête utilisée. Une fois le chemin trouvé, il suffit de partir du nœud final, de regarder l'arête utilisée terminant par ce nœud, et de créer la tournée correspondante. Cette étape est répétée jusqu'à ce que toutes les tournées nécessaires soient créées (figure 3.14). Le pseudo-code complet est disponible en annexe, à la page 176.

Cette méthode est optimale, et de complexité en $O(nb)$ (où $b := \left\lfloor \frac{charge_{max}}{demande_{min}} \right\rfloor$ correspond au nombre maximum de clients pouvant être livré par un véhicule) et trouve un découpage optimal pour les problèmes sans limite de véhicules. Lorsque le nombre de véhicules est limité, on attribue à chaque sous chemin, depuis le départ, un triplet {coût, nombre de véhicules utilisés, nœud de départ}. Lors de l'étude d'une nouvelle arête, chaque triplet devra être pris en compte. Lors de la création d'un nouveau triplet, celui-ci sera comparé aux autres triplets du nœud d'arrivée. S'il en existe entraînant un coût inférieur, en utilisant moins de véhicules (pour chacune des catégories), alors le triplet n'est pas ajouté à la liste des triplets du nœud d'arrivée. Pour conserver la complexité polynomiale, il est possible de limiter le nombre de triplets associés à chaque nœud, cependant, on perd ainsi l'optimalité, voire même la faisabilité de la séparation.

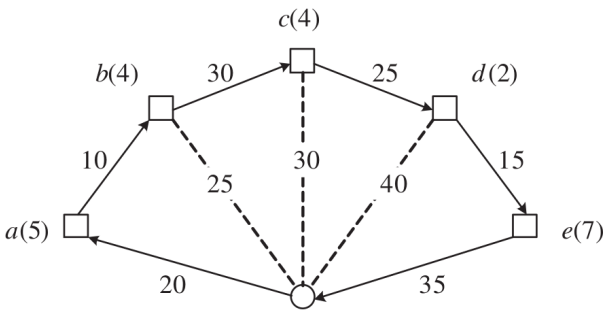


FIGURE 3.12 – Exemple de tour géant

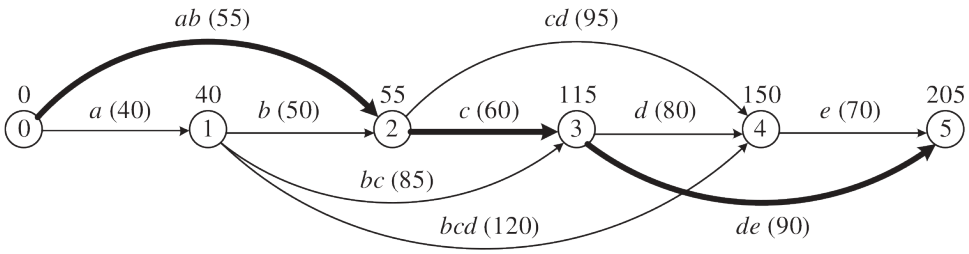


FIGURE 3.13 – Exemple de graphe de plus court chemin utilisé

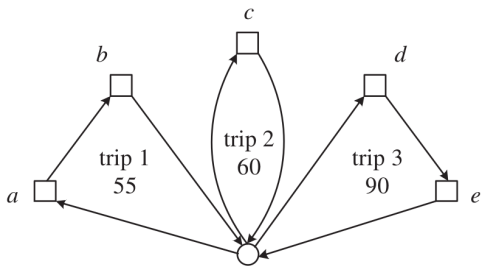


FIGURE 3.14 – Solution obtenue après le split

Voisinages supplémentaires :

Les chaînes d'éjection Lors d'une recherche locale, il est fréquent de voir des mouvements intéressants être interdits car ne respectant pas certaines contraintes. Si celui-ci réduit considérablement le coût de la solution, il est malgré tout tentant d'exécuter ce mouvement. Pour cela, il est par exemple possible d'utiliser les chaînes d'éjections. Ces chaînes représentent une suite de mouvements dont le premier n'est pas réalisable. Les mouvements suivants sont là pour réparer la contrainte non respectée. Le mouvement s'arrête lorsqu'une nouvelle solution réalisable est obtenue, ou lorsqu'un nombre conséquent de réparations infructueuses ont été testés.

La figure 3.15 montre un exemple où les chaînes d'éjections sont nécessaires : imaginons que la capacité des véhicules soit limitée, au point de n'autoriser que 5 clients par tournée. Dans la figure initiale, le véhicule bleu fait un détour en début de tournée pour aller livrer le client 4. Pendant ce temps, le véhicule rouge, passe près du véhicule 4 en parcourant l'arête 5-6. Il semble donc intéressant de changer la tournée d'affectation du client 4. Cependant, le véhicule rouge livre déjà 5 clients, et ne peut donc pas livrer le client 4. La modification de la tournée va se faire en deux étapes :

1. on affecte le client 4 à la tournée rouge
2. on cherche un client à enlever de la tournée rouge

Lors de la deuxième étape, le client 9 a été enlevé de la tournée rouge, pour être affecté à la tournée violette. Celle-ci livre désormais 5 clients, ce qui respecte sa contrainte de capacité. La nouvelle solution est donc réalisable, l'algorithme des chaînes d'éjections s'arrête.

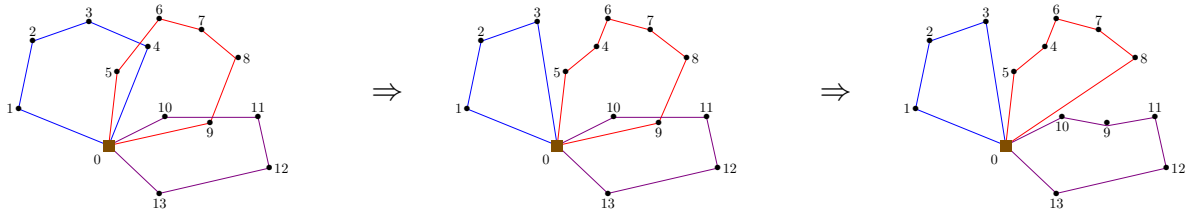


FIGURE 3.15 – Exemple de chaîne d'éjection

3.2.3 Poursuivre la recherche

Une fois les différents voisinages utilisés, le solveur obtient une solution présente dans un minimum local. Cette partie va présenter deux méthodes utilisées dans le solveur pour poursuivre la recherche de solutions intéressantes.

— La méthode GRASP, qui multiplie les solutions de départ,

— La recherche tabou, qui se permet de se déplacer même lorsque les solutions voisines sont de moins bonne qualité.

Ces méthodes augmentent le nombre de solutions observées, et donc le temps total d'utilisation du solveur.

Greedy Randomized Adaptive Search Procedure (GRASP)

La première méthode pour augmenter les chances d'obtenir une bonne solution est de multiplier les solutions de départ. En effet, partir de plusieurs solutions différentes augmente la probabilité de tomber sur un minimum local de bonne qualité (voir figure 3.16).

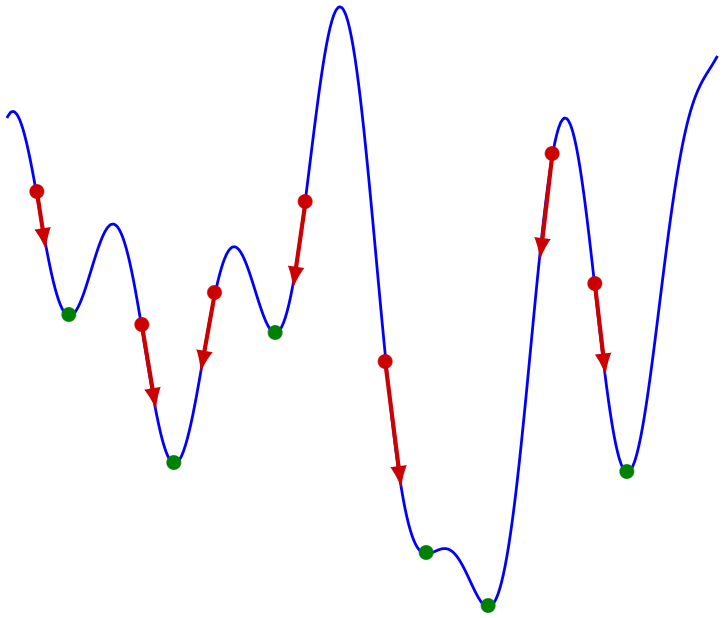


FIGURE 3.16 – Exemple de GRASP

Le GRASP est donc une méthode en deux étapes :

1. Générer une population de solutions initiales,
2. Faire une recherche locale à partir de chaque solution.

Pour cela, il est nécessaire de générer rapidement un ensemble de solutions différentes. Or, la plupart des algorithmes de création de solutions sont gloutons, donc déterministes.

Le GRASP parvient à générer une population en ajoutant une part d'aléatoire : considérons l'algorithme de Clarke and Wright, qui choisit de fusionner le couple (i, j) de tournées, en fonction de leur savings $s_{i,j}$. À chaque étape, la version gloutonne de l'algorithme choisit le couple $\text{argmax}_{(i,j)} s_{i,j}$. Dans la version randomisée, l'algorithme choisit un des couples admissibles dont le saving est supérieur ou égal à $\alpha * \max_{(i,j)} s_{i,j}$. Le coefficient $\alpha \in]0; 1[$ est le coefficient de randomisation :

— Une valeur de α proche de 0 autorise la plupart des fusions de tournées, et va privilégier la diversification des solutions à leur qualité.

— Une valeur de α proche de 1 va au contraire n'autoriser que les meilleurs mouvements.

— Si $\alpha = 1$, alors on obtient l'algorithme de Clarke and Wright non randomisé.

Recherche Tabou

La méthode GRASP présentée précédemment permet certes de se donner une seconde chance, après une première descente, mais elle présente l'inconvénient de recommencer de zéro à chaque fois. Au lieu de partir d'une nouvelle solution, la méthode de recherche tabou va essayer de poursuivre les explorations. Pour cela, la méthode tabou cherche à aller dans la meilleure direction, même si celle-ci n'est pas améliorante. Dans le cas où une solution améliorante a été obtenue, cela revient à faire une recherche locale classique.

Si aucune solution améliorante n'est accessible dans le voisinage, la méthode tabou, au lieu de s'arrêter, va poursuivre sa progression, en cherchant la "*moins mauvaise*" solution possible.

Cependant, en procédant ainsi, il se peut que l'on arrive à la situation suivante, représentée par la figure 3.17 :

Soit x_0, la solution initiale (obtenue par exemple avec l'algorithme de Clarke and Wright). Au bout d'un certain nombre d'itérations, on obtient une solution x_1, réalisable, qui est un minimum local. L'algorithme de recherche tabou ne s'arrête pas et décide de se déplacer en x_2, la moins mauvaise des solutions voisines. Lors de l'étape suivante, aucune solution voisine de x_2 est améliorante, à l'exception de x_1. L'algorithme se déplace donc en x_1. Or, on a vu précédemment que la meilleure solution du voisinage de x_1 est x_2. L'algorithme va donc se déplacer en x_2 à l'étape suivante. On obtient ainsi une boucle infinie où l'algorithme ne va explorer que les solutions x_1 et x_2.

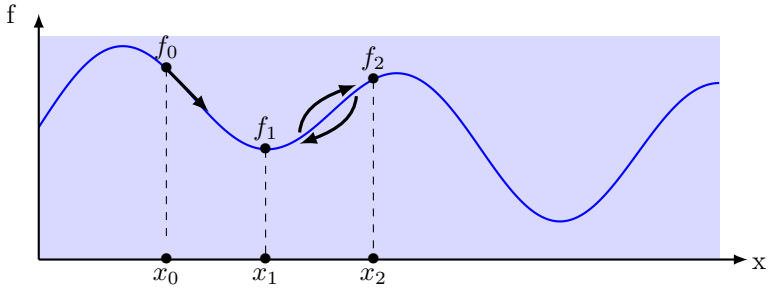


FIGURE 3.17 – Exemple de boucle infinie possible

Pour éviter cette situation, la méthode Tabou stocke les k dernières solutions. Si la solution la plus intéressante fait partie de ces k solutions, la méthode tabou va l'ignorer, et chercher la deuxième solution la plus intéressante. En pratique, stocker et comparer des solutions entières nécessite trop de calculs. Au lieu de stocker des solutions, les solveurs

vont plutôt interdire certains mouvements.

Par exemple, si pour passer de la solution X_0 à la solution X_1, il suffit de modifier l'affectation du client 5, on peut interdire de modifier ce client pendant les k prochaines itérations.

La valeur de k est un paramètre à bien doser : une valeur trop petite augmente le risque de tomber sur une solution déjà étudiée, et donc de boucler. Au contraire, une valeur trop grande de k risque de réduire de manière significative les mouvements autorisés, et donc de réduire l'espace de recherche.

L'algorithme s'arrête au bout d'un certain nombre d'itérations infructueuses.

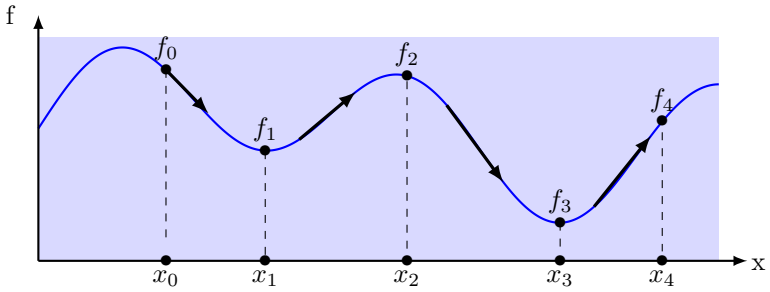


FIGURE 3.18 – Exemple de poursuite de recherche locale par la méthode Tabou

Dans l'exemple ci-dessus (figure 3.18), on part de la solution x_0. Comme il existe une direction de descente à partir de x_0, les premières itérations de la recherche tabou sont similaires à une recherche locale classique. Après plusieurs mouvements, on finit par obtenir la solution x_1. Au lieu de s'arrêter, le solveur va continuer d'explorer l'ensemble des solutions.

Dans ce schéma simplifié, il n'y a que deux directions possibles : vers la gauche, ou vers la droite. La première direction étant celle menant aux solutions déjà explorées, seule la direction vers la droite est autorisée. Cette direction n'est malheureusement pas améliorante, on arrive, après plusieurs mouvements, à la solution x_2, qui est de qualité moindre par rapport à x_1. Cependant, depuis x_2, une direction de descente est à nouveau accessible. Après plusieurs mouvements, le solveur obtient la solution x_3, qui est de bien meilleure qualité que x_1. L'algorithme poursuit ensuite la recherche, jusqu'à épuiser son quota de mouvements non améliorants, et s'arrête en x_4. Le solveur finit par retourner x_3, la meilleure solution observée.

3.2.4 Réduction de la zone de recherche

Dans la session précédente, nous avons vu que chaque recherche locale étudie $O(n^2)$ solutions voisines possibles. Cette complexité peut sembler faible, mais elle implique une nette augmentation du temps de calcul de tous les voisins pour les instances de grande taille (plusieurs centaines de clients) et très grande taille (plus de 1000 clients). Cependant, en observant attentivement, il est possible de se rendre compte que certains mouvements sont inutiles : dans le cas d'une livraison sur l'ensemble du territoire français, il paraît peu probable de faire une livraison à Brest juste après une livraison à Marseille. Il est donc courant d'éliminer une partie des arêtes de longue taille, pour réduire le nombre de mouvements possible, et donc le temps d'exécution total.

Une étude récente de Arnold *et al.* (2019a) nous permet d'en savoir plus sur cette méthode de simplification de graphes. Les auteurs ont étudié plusieurs instances de très grande taille, pour lesquelles ils n'ont gardé qu'une partie des arêtes. Plus précisément, pour chacun des n clients, ils n'ont conservé que les k arêtes les plus courtes, partant de ce client. Le nombre d'arêtes étudiées est donc passé de n^2 à $k \times n$: la taille du graphe a été linéarisée. Pour connaître la valeur adéquate de k, les auteurs ont fait une étude statistique sur plusieurs instances de grande taille. Ils ont étudié l'évolution de la qualité des solutions dans une recherche tabou (voir section TABOU pour plus d'informations) selon plusieurs valeurs de k.

Notons le rang de j (par rapport à i) la position de l'arête ij dans le classement croissant des arêtes partant de i. L'étude démontre que pour les solutions de bonne qualité des 100 instances utilisées par Uchoa *et al.* (2017), plus de 99% des nœuds sont connectés par des voisins de rang inférieur à 50. Selon le voisinage utilisé (intra-route ou inter-route), et le type d'instance, les auteurs suggèrent une valeur de k comprise entre 20 et 50.

La figure 3.19 montre l'effet de la réduction du graphe à l'échelle d'un client. Sur cet exemple, seules les 4 arêtes les plus courtes partant du client étudié sont gardées. La figure 3.20 montre la réduction du nombre d'arêtes sur un graphe complet, composé de 10 nœuds, pour $k = 4$.

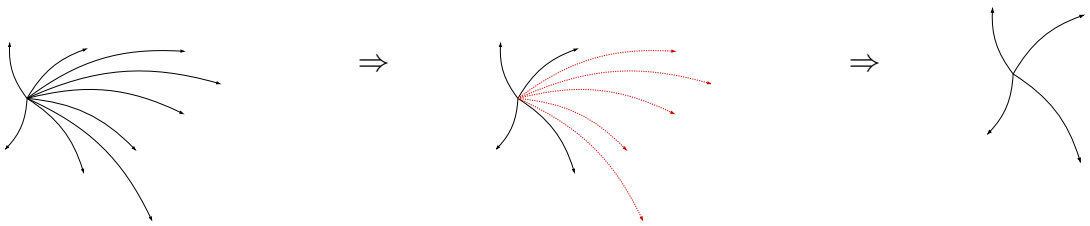


FIGURE 3.19 – Exemple de réduction locale des arêtes

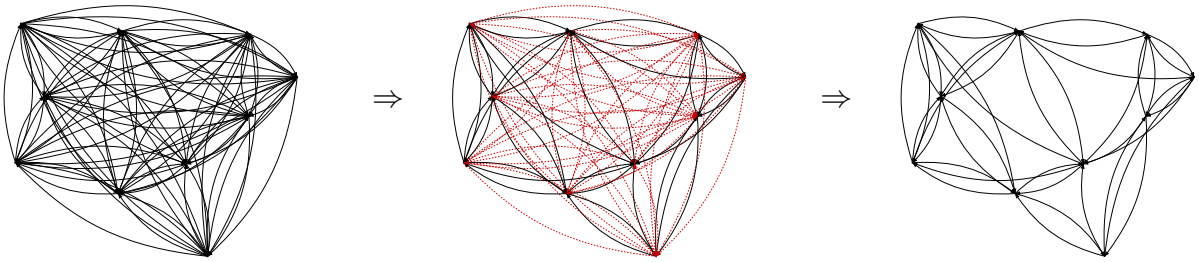


FIGURE 3.20 – Exemple de réduction globale des arêtes

Remarque : Le graphe ainsi obtenu perd toute propriété de symétrie. Soit un problème à trois nœuds, A, B et C représenté par la Figure 3.21. Le voisin de rang 1 du nœud A est B, celui de B est C et inversement. Ainsi, A est connecté à B, mais B n'est pas connecté à A, le graphe est donc assymétrique.

FIGURE 3.21 – Exemple de perte de symétrie

La manière naïve pour obtenir facilement ce sous-graphe est de trier pour chaque client l'ensemble de ses arêtes associées, et de ne garder que les k premières arêtes. Cette façon de faire à un temps d'exécution en $O(n^2 log_2(n))$, ce qui n'est pas utilisable en pratique. L'astuce permettant de rendre ce pré-traitement utilisable sur des instances de très grande taille se trouve en annexe de ce document, à la page 177.

3.3 Descriptif de l'ensemble de la méthode

Cette section va indiquer comment les notions acquises précédemment interagissent entre elles, apportant une meilleure compréhension du fonctionnement global de RADOS.

3.3.1 Solutions initiales

Comme indiqué précédemment, une population de N_{genere} solutions initiales est générée par un algorithme du type GRASP, appliqué sur l'algorithme de Clarke and Wright. Cependant, toutes les solutions générées ne sont pas intéressantes. Plus une solution initiale est de mauvaise qualité, plus le nombre de mouvements moyens nécessaires pour arriver à un minimum local est élevé. De plus, la probabilité pour que la solution obtenue après recherche locale soit intéressante est plus faible.

C'est pour toutes ces raisons qu'une phase supplémentaire de sélection des élites intervient. Dans cette phase, seules les N_{elite} meilleures solutions sont gardées. Les nombres N_{genere} et N_{elite} étant des paramètres du solveur, où $N_{elite} \leq N_{genere}$.

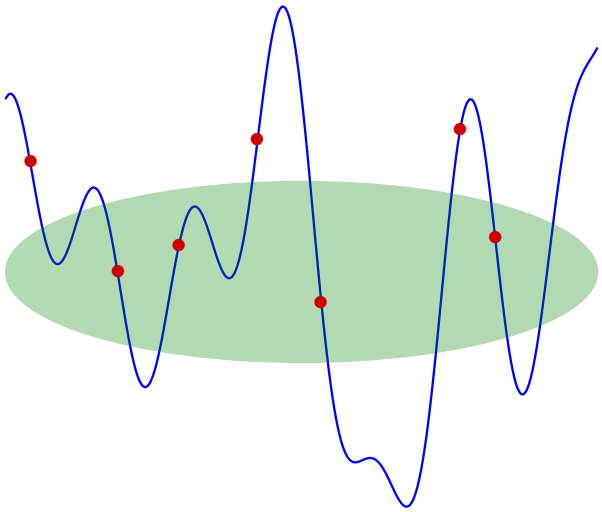


FIGURE 3.22 – Sélection des 4 meilleurs solutions initiales

3.3.2 Recherche locale multiple et tabou

Nous avons vu que l'idée principale du solveur est d'améliorer chacune des solutions, via une recherche locale. De nombreux voisinages sont utilisés, afin d'augmenter

les chances d'obtenir une solution intéressante. Chaque voisinage est utilisé dans une recherche tabou. Celle-ci s'arrête après un nombre fixé d'itérations sans obtenir de nouvelle meilleure solution. Une fois arrêtée, la méthode tabou retourne la meilleure solution obtenue, puis une nouvelle recherche est lancée, avec un nouveau voisinage. L'algorithme s'arrête lorsque tous les voisinages ont été appliqués sur une solution par recherche tabou, sans succès d'amélioration.

La figure 3.23 montre un exemple de chemin parcouru dans l'espace des solutions, lors de l'utilisation de deux voisinages différents. Dans cet exemple, la solution initiale x_0 est améliorée par le mouvement 1, jusqu'à être bloquée au niveau de la solution x_1. Cette solution est elle-même améliorée par une recherche tabou utilisant le mouvement 2, jusqu'à être bloquée en x_3. Cette fois-ci, on effectue une nouvelle recherche tabou, via l'utilisation du premier voisinage. Et ainsi de suite, jusqu'au moment où aucun des voisinages associés aux mouvements 1 et 2 ne contiennent de solutions de meilleure qualité que la solution courante.

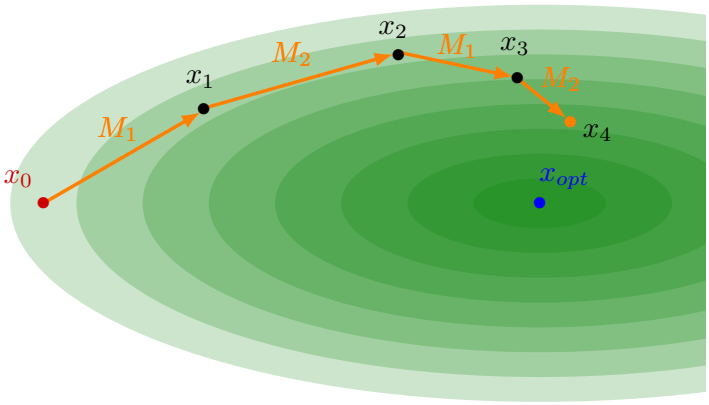


FIGURE 3.23 – Exemple de chemin parcouru dans des méthodes à voisinages multiples

Lorsque plusieurs voisinages sont implémentés, il faut définir l'ordre dans lequel ceux-ci doivent être exécutés. La figure 3.24 illustre le fait que le chemin parcouru dans l'espace des solutions est fortement dépendant de la séquence d'utilisation des voisinages. Il n'existe à notre connaissance aucune règle explicite définissant le voisinage à utiliser en fonction de la solution courante. Il a donc été décidé de choisir aléatoirement l'ordre d'utilisation des voisinages, au début de la recherche locale. Cette séquence est modifiée lors de l'appel d'une nouvelle solution à améliorer.

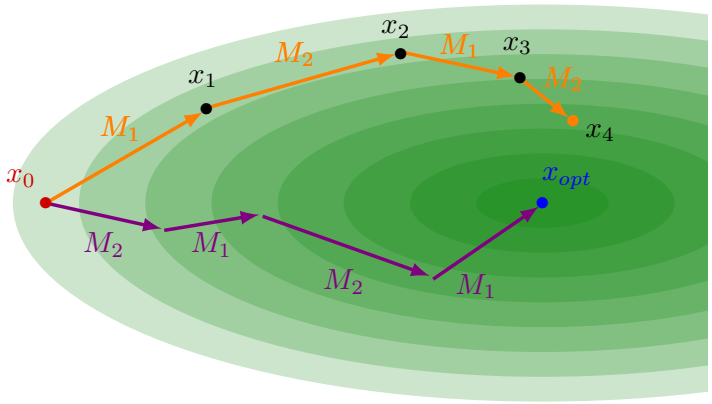


FIGURE 3.24 – Les solutions parcourues dépendent de l'ordre dans lesquels les voisinages sont exécutés

De plus, plusieurs séquences peuvent être testées pour une même solution. Dans l'exemple représenté dans la figure 3.24, cela revient à parcourir le chemin orange (séquence de voisinage 1-2) puis, le chemin violet (séquence de voisinage 2-1).

3.3.3 Résumé

Le solveur est donc basé sur l'amélioration de solutions initiales. Ces solutions sont filtrées parmi celles obtenues avec un GRASP basé sur l'algorithme de Clarke and Wright. Chacune de ces solutions est améliorée par une succession de recherches locales, basées sur des mouvements différents, et exécutées dans une séquence aléatoire. Ces séquences sont modifiées à chaque nouvelle solution à améliorer. Chaque solution peut être testée sur plusieurs séquences de voisinages. Enfin, chaque recherche locale est complétée par une recherche tabou, autorisant un certain nombre d'itérations infructueuses.

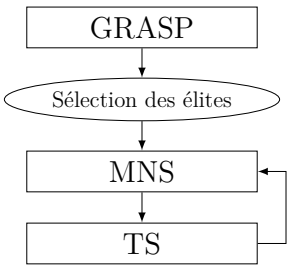


FIGURE 3.25 – Organigramme du solveur RADOS

3.4 Un solveur utilisé en industrie

Dans le cadre de la thèse, nous avons eu l'opportunité de collaborer avec la société Mapotempo, et plus particulièrement avec Gwénaël Rault, doctorant dans cette société. Mapotempo est une société de service d'une vingtaine d'employés, elle édite des solutions de cartographie, de planification et d'optimisation de tournées.

L'application Mapotempo Web développée par cette société permet de représenter aisément la position des noeuds à livrer sur une carte ainsi que d'y renseigner leurs demandes, créneaux horaires et diverses contraintes supplémentaires. L'application permet également de renseigner la composition de la flotte, leurs caractéristiques et leurs capacités d'emport. (voir figure 3.26).

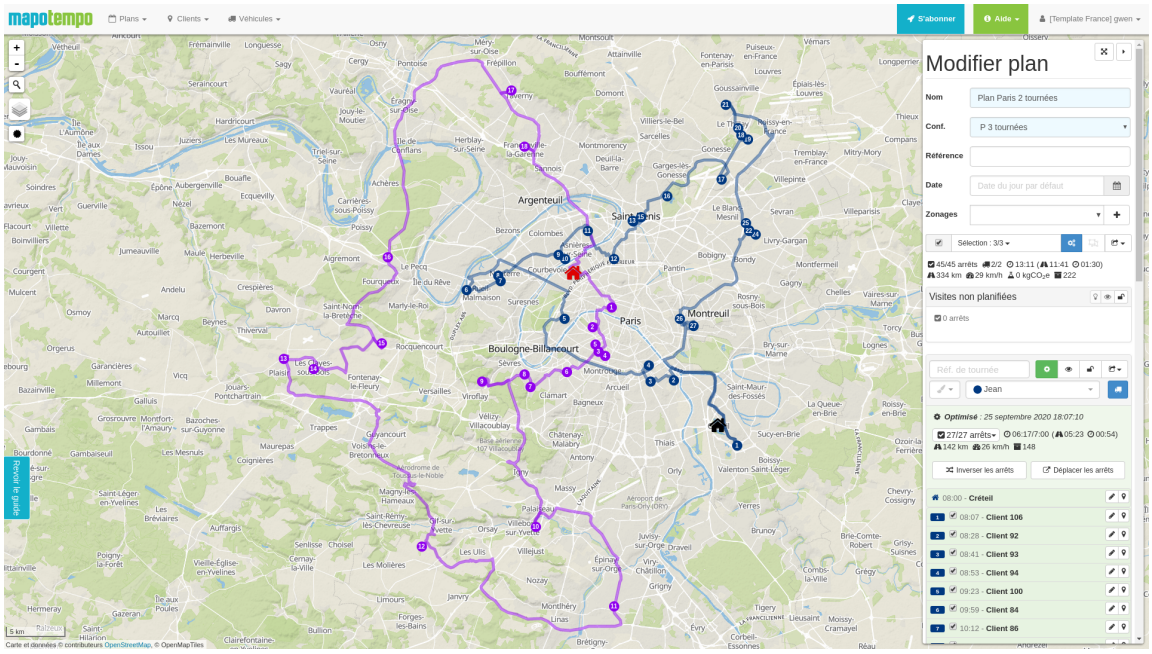


FIGURE 3.26 – Interface de Mapotempo Web

Cette collaboration a mené à l'intégration du solveur RADOS au sein du projet Optimizer API (voir figure 3.27), qui est dédié à la résolution des problèmes de tournées de véhicules. Le projet traite les instances qui lui sont fournies et redirige vers la méthode de résolution la plus adaptée en fonction contraintes des clients. Dans certains cas de figures RADOS est utilisé pour générer une première solution avant d'appeler d'autres solveurs (Rault *et al.*, 2019).

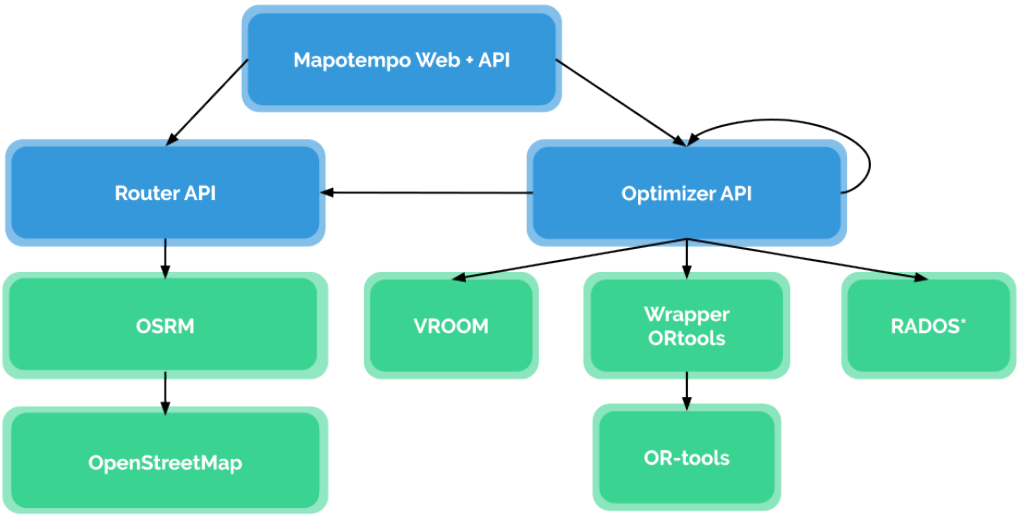


FIGURE 3.27 – Ecosystème de projets utilisés par Mapotempo

Une plate-forme open source est en cours de création, grâce à l'aide de Pierre Bomel (ingénieur de recherche à l'UBS) et Gwénaël Rault, permettant la génération d'instances basées sur les données cartographiques fournies par le projet OpenStreetMap (OSM). Un ensemble de clients (commerces, restaurants, etc.) est extrait de ces données et projetées ensuite sur le réseau routier. Certaines de ces instances seront testées dans le chapitre 5. Le générateur se base entre autre sur l'appel à l'API Overpass pour obtenir l'ensemble des clients et dépôts. Une interface web sera bientôt disponible pour manipuler cette API, elle est représentée par la figure 3.28.

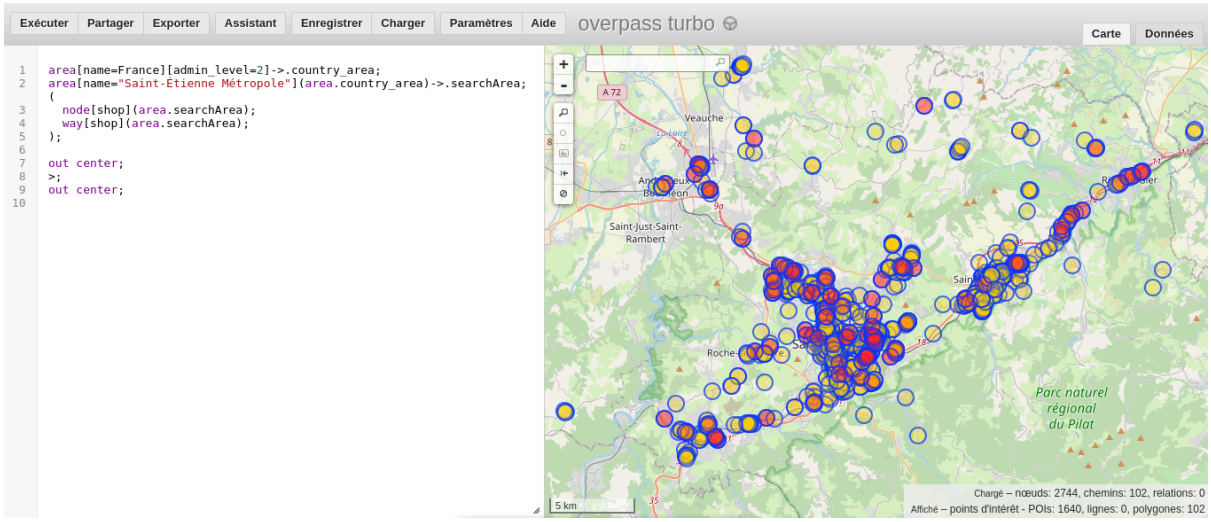


FIGURE 3.28 – Exemple d'appel pour générer les clients
(© les contributeurs OpenStreetMap)

Une fois l'ensemble des clients obtenu, pour chaque catégorie de véhicule demandé, la matrice de distance est générée à partir d'itinéraires projetés sur le réseau routier et ce entre chaque paire de nœuds, avec le profil de véhicule demandé. Le calcul d'itinéraire peut se faire en minimisant la distance des trajets ou en minimisant le temps de ces trajets. Les clients sont également catégorisés en fonction de la densité de la zone dans laquelle ils sont localisés (faible densité, forte densité et très forte densité).

3.5 Performances

Cette section montre l'efficacité du solveur sur différents jeux d'instances. Celui-ci a en effet été implémenté dans le but d'obtenir un outil de résolution robuste, capable de trouver des solutions de qualité acceptable sur une grande variété d'instances. Afin de tester les limites du solveur, celui-ci a été exécuté sur deux catégories d'instances jugées difficiles. Le premier ensemble d'instance a été créé par F. Arnold et K. Sörensen et permet de tester la rapidité des solveurs sur des problèmes de très grande taille. Le deuxième jeu d'instance, créé par Duhamel, Lacomme et Prodhon contient des problèmes à flottes hétérogènes non euclidiens, basés sur de vraies données géographiques.

Toutes les expérimentations ont été effectuées sur un DEll Latitude 7480, avec 16 Go de RAM, et un processeur i7-7600U cadencé à 2.80GHz. Le solveur a été implémenté en C++.

3.5.1 Instances XXL

La plupart des solveurs utilisant des algorithmes de complexité quadratique, voire supérieurs, ont généralement du mal à passer à l'échelle. Ainsi, encore récemment (Uchoa *et al.*, 2017), il était nécessaire d'attende plusieurs heures avant d'obtenir des solutions de bonne qualité pour des problèmes comportant plusieurs centaines de clients. En utilisant des astuces de réduction de l'espace de recherche, dont celle décrite en section 3.2.4, Arnold *et al.* (2019a) ont réussi à résoudre des instances de plusieurs dizaines de milliers de clients en quelques dizaines de minutes.

Ils ont plus précisément utilisé une méthode de recherche tabou à voisinage multiple, qui ne s'arrête qu'après une limite de temps définie. Les auteurs comparent ensuite les performances de leur solveur (Arnold *et al.*, 2017), nommé A-G-S, selon deux paramétrages de l'algorithme.

1. Une version longue s'arrêtant au bout d'un temps $T = 5 * \frac{N}{1000}$ minutes.
2. Une version courte s'arrêtant au bout d'un temps $T = \frac{N}{1000}$ minutes.

Où N est le nombre de clients.

FIGURE 3.29 – Exemple de solution réalisable d'une instance XXL (sans les arêtes extrêmes)

Le tableau 3.1 établit un comparatif entre la qualité des solutions obtenues par les solveurs RADOS et A-G-S (version courte), et leur temps d'exécution. La qualité des solutions est évaluée de manière relative, en se comparant aux solutions obtenues par la version longue du solveur A-G-S.

En observant les performances de RADOS, on peut remarquer que les solutions obtenues ont un écart moyen de 2.59% par rapport aux meilleures solutions connues. Ce qui est respectable pour des instances de cette taille, même si la version courte du solveur A-G-S est capable de trouver des solutions d'une qualité supérieure, avec un écart moyen de 1.14%.

Instances	Nombre de clients	A-G-S (version courte)		RADOS	
		Écart moyen (%)	Temps CPU (s)	Écart moyen (%)	Temps CPU (s)
L1	3.000	**1.36**	180	1.67	**1.91**
L2	4.000	**2.62**	240	5.15	**2.47**
A1	6.000	**0.22**	360	1.67	**7.20**
A2	7.000	**1.68**	420	3.69	**5.59**
G1	10.000	**0.35**	600	1.44	**12.46**
G2	11.000	**1.27**	660	2.99	**17.94**
B1	15.000	**0.77**	900	2.05	**24.75**
B2	16.000	**1.89**	960	2.79	**16.86**
F1	20.000	**0.42**	1200	1.6	**57.90**
F2	30.000	**2.04**	1800	2.88	**108.46**
Moyenne		**1.14**	**732**	**2.59**	**25.55**

Tableau 3.1 – Comparaison de l'A-G-S et de RADOS sur des instances XXL

Cependant, si on observe en parallèle les temps d'exécution, RADOS gagne en intérêt. En effet, bien qu'il ait un écart supérieur de 1.5% par rapport au solveur A-G-S, il met en moyenne presque 30 fois moins de temps. Ainsi, le solveur RADOS est capable de trouver en un temps record, des solutions de qualité plus qu'acceptable sur des instances de très grande taille.

3.5.2 Instances DLP

Une autre variante d'instances réduisant particulièrement les performances des solveurs, sont les instances à flottes hétérogènes. En effet, sur ce type d'instances, les véhicules disponibles font partie de plusieurs catégories. Chaque catégorie pouvant contenir sa propre matrice de distance, capacité, ainsi qu'un coût d'utilisation, indépendant de la distance parcourue par celui-ci. Ces instances compliquent la décision du solveur, car en plus de décider de l'affectation clients/véhicules, il convient désormais de choisir la bonne catégorie de véhicule à attribuer à un groupe de clients.

L'ensemble d'instances jugé le plus difficile est l'ensemble DLP, (pour *Duhamel, Lacomme et Prodhon*, le nom de leurs auteurs), et basé sur de vraies données cartographiques (Duhamel *et al.*, 2011). En effet, en plus de posséder jusqu'à 8 catégories différentes de véhicules, et plusieurs centaines de clients, ces instances utilisent les distances réelles et donc ne respectent pas l'inégalité triangulaire, rendant inutiles certaines astuces.

En raison de leur complexité, seule une partie de ces instances a été résolue avec une preuve d'optimalité (Sadykov *et al.*, 2017). Cet ensemble d'instances a donc été séparé en deux catégories :

— Les instances les plus modestes, dont on connaît la valeur de la solution optimale. Ces instances sont dites de catégorie *A*.

— Les instances les plus complexes, sans preuve d'optimalité. Ces instances sont dites de catégorie *B*.

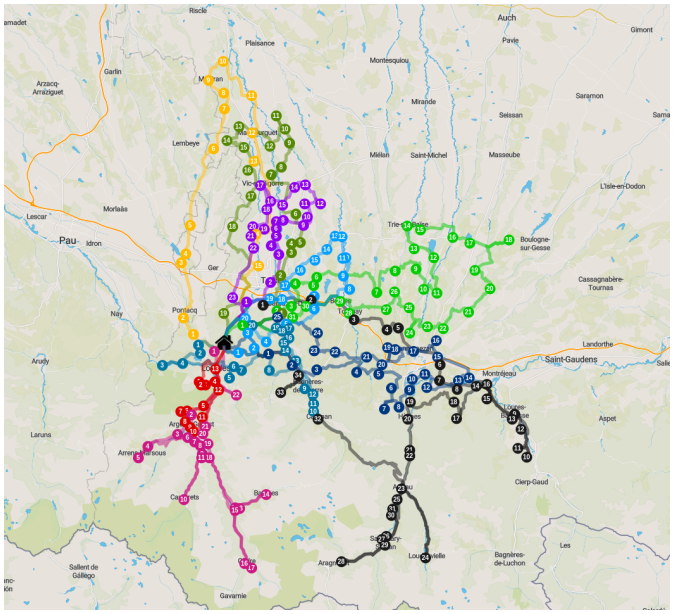


FIGURE 3.30 – Exemple d'instances de type DLP
(© les contributeurs OpenStreetMap)

Le tableau 3.2 résume les caractéristiques des deux catégories d'instances, ainsi que les performances du solveur RADOS sur ces instances. La qualité des solutions est à nouveau évaluée en fonction de l'écart relatif par rapport au meilleur coût trouvé dans la littérature, avec ou sans preuve d'optimalité. Les performances détaillées, instance par instance sont présentées en annexe, à la page 179.

Catégorie de l'instance	Nombre de clients	Nombre de véhicules	Écart moyen (%)	Temps CPU moyen (s)
Catégorie A	20 to 186	3 to 8	6.56	2.61
Catégorie B	77 to 256	2 to 8	7.78	5.45

Tableau 3.2 – Performances on DLP instances

Le solveur RADOS obtient donc des solutions avec une qualité moyenne supérieure à 6% quelle que soit la catégorie de l'instance. Celui-ci ne rivalise donc pas avec la littérature si on se focalise uniquement sur la qualité des solutions. Cependant, ces solutions sont obtenues en quelques secondes seulement, alors qu'il faut plusieurs dizaines de secondes, voire plusieurs minutes pour obtenir des solutions de bonne qualité dans la littérature (Penna *et al.*, 2019). Plusieurs heures ont parfois été nécessaires pour résoudre avec preuve d'optimalité certaines instances de catégorie A (Sadykov *et al.*, 2017).

3.6 Conclusion

Bien que générique, le solveur RADOS est capable d'obtenir des solutions tout à fait acceptables en très peu de temps pour un nombre conséquent de variantes, y compris sur des instances jugées difficiles. Cette performance a permis de l'intégrer au sein d'un logiciel professionnel, grâce à la collaboration avec la société Mapotempo. Seul un certain nombre de variantes a été testé, mais il semblerait que l'adaptation du solveur RADOS avec d'autres variantes ne présente a priori pas de difficultés particulières. En particulier, le fait de minimiser le coût des arêtes utilisées (en plus des éventuels coûts fixes des véhicules) sans se préoccuper de la topologie de la matrice associée permet de considérer un grand nombre de variantes sans distinction. En effet, le coût de l'arête représentant le trajet du chemin de i à j peut aussi bien représenter une distance à minimiser (symbolisant la consommation de carburant), le temps de trajet (réduction du coût horaire), le quantité de CO2 émise (réduction de la pollution), ou tout autre combinaison linéaire de ces facteurs.

En conclusion de ce chapitre, nous avons répondu à la question de recherche 1 : il est effectivement possible de disposer d'un solveur adaptatif et capable de résoudre de très grandes instances. Nous verrons en section 5 que ce solveur est également adapté pour être combiné avec des méthodes d'analyse statistique.

Chapitre 4

Caractérisation descriptive d'une solution

Dans ce chapitre, nous allons déterminer ce qui constitue une bonne solution. Pour être décrite, une solution a besoin de caractéristiques claires et explicatives. Dans notre recherche de caractéristiques intéressantes, nous commencerons par l'utilisation de méthodes naïves, dont les limites seront vite atteintes. Afin d'utiliser des méthodes plus robustes, nous allons introduire le domaine de la fouille de données. Ces outils vont nous permettre d'étudier des caractéristiques présentes dans la littérature, nous permettant ainsi de mettre en place une méthodologie testant l'efficacité de ces caractéristiques.

Sommaire

4.1 Vers un besoin d'outils statistiques

Avant de chercher à utiliser un éventail complet d'outils de fouilles de données, nous allons voir s'il est possible de se contenter de méthodes naïves.

4.1.1 Étude visuelle des solutions

En travaillant sur des instances gagnant en réalisme, il devient évident que notre perception intuitive de la notion de bonne solution est éloignée de la réalité. En observant les solutions optimales du problème du voyageur de commerce sur des instances euclidiennes, nous pouvons observer que celles-ci ne contiennent aucun croisement. Comme mentionné dans le chapitre précédent, cette absence obligatoire de croisement (dont la démonstration est disponible en annexe page 175) a inspiré de nombreux voisinages, dont le 2-opt.

Cependant, en passant du voyageur de commerce aux problèmes de tournées, cette observation ne tient plus. En effet, si prise individuellement, chaque tournée correspond à un voyageur de commerce, et ne contient pas de croisement, il est fréquent de voir deux tournées se croiser entre elles.

L'hypothèse *une solution optimale ne contient aucun croisement* se complexifie et devient *dans une solution optimale, aucune tournée ne contient de croisement.*

Les choses se compliquent à nouveau en traitant des problèmes issus de milieux urbains. En effet, dans la circulation urbaine, il est rare que le trajet entre deux clients soit une simple ligne droite. De plus, la vitesse moyenne sur chaque portion de route est différente, en raison des limitations légales, du trafic routier, etc. Une instance en milieu urbain ne respecte donc pas l'inégalité triangulaire, et l'hypothèse précédente n'est plus prouvée.

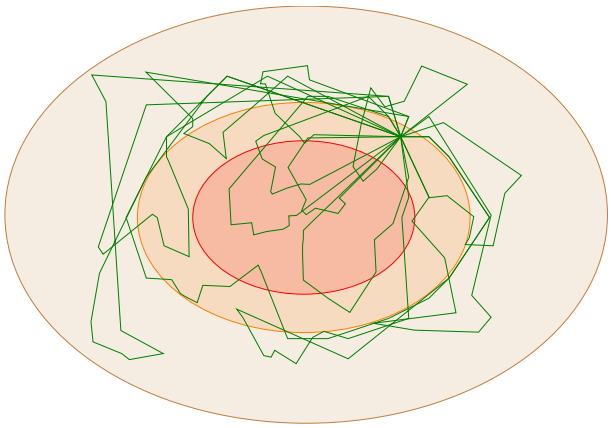


FIGURE 4.1 – Exemple d'instance non euclidienne générée

Afin de chercher des règles pouvant caractériser des solutions réalistes, un générateur d'instances non euclidiennes a été implémenté. Chaque instance est censée simuler une ville. Celle-ci est constituée de 3 zones :

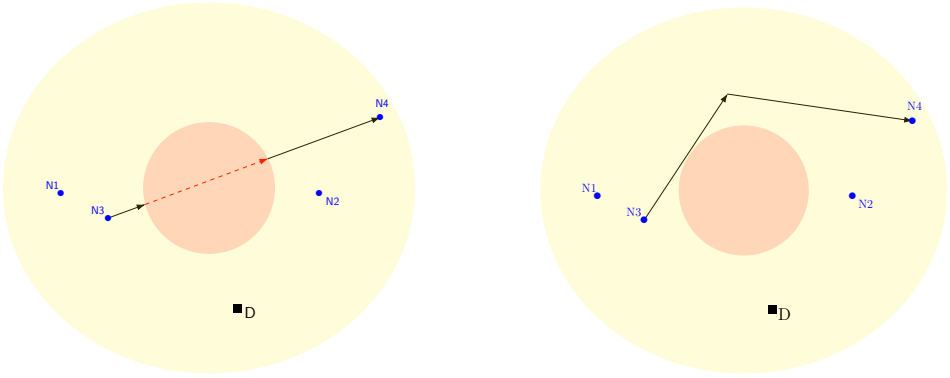
— l'hypercentre urbain, en rouge sur la figure 4.1. C'est la zone la plus dense de la

ville, les véhicules s'y déplaçant subissent une forte pénalité, proportionnelle à la distance parcourue.

— le centre urbain (en orange sur la figure). Cette zone représente des endroits denses, mais plus accessible que l'hypercentre. Une pénalité faible y est appliquée.

— les zones à faible densité (marron clair sur la figure), représentant les zones où la circulation est fluide.

Avec cette topologie, il est parfois plus intéressant de faire une plus grande distance géographique, pour réduire le temps total de parcours. La figure 4.2(a) illustre une situation où, pour livrer le client N_4 après le client N_3, la ligne droite ne semble pas être le plus court chemin, si la pénalité dans la zone rouge est trop importante. Cette situation peut représenter un usager qui éviterait une zone congestionnée.



(a) Chemin euclidien (b) Chemin réel

FIGURE 4.2 – Comparaison des chemins euclidiens et réels

Pour permettre assez facilement aux véhicules de faire un détour, des nœuds intermédiaires ont été ajoutés. Un véhicule peut visiter un ou plusieurs nœuds entre deux clients, comme illustré par la figure 4.2(b).

Reprenons l'exemple de la figure 4.1, en n'affichant que certaines tournées, pour plus de visibilité.

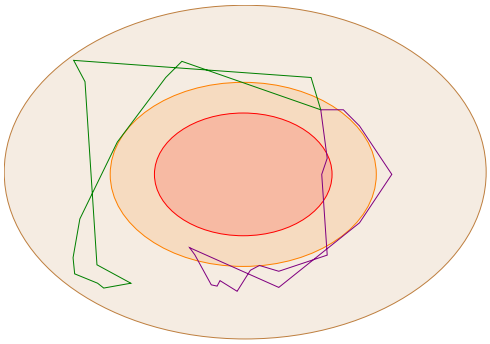


FIGURE 4.3 – Exemple de croisements intra-route

Dans la Figure 4.3, les deux tournées affichées comportent des croisements inter-route. Celles-ci ont été obtenue avec le solveur RADOS défini au chapitre précédent. Il n'y a aucune preuve d'optimalité, mais chaque solution retournée a été le point de départ d'une recherche locale utilisant le 2-opt, qui aurait dû supprimer ces croisements dans le cas euclidien. Ceux-ci sont donc nécessaires pour réduire la durée totale de la tournée. En observant plus attentivement, on peut remarquer que les trajets des tournées frôlent les zones denses, mais n'y entrent que pour livrer un client. En appliquant un 2-opt au niveau des différents croisements, la distance parcourue dans les zones denses augmente.

L'hypothèse formulée plus haut est donc effectivement fausse sur des instances non euclidiennes, même si celles-ci sont encore simples par rapport à la réalité géographique.

Cependant, peut-on quantifier le nombre de croisements nécessaires ? Ici il convient de définir le terme de croisement : parle-t-on de croisement sur le chemin euclidien (figure 4.4(a)), ou sur le chemin réel (figure 4.4(b)) ?

(a) Chemin euclidien (b) Chemin réel

FIGURE 4.4 – Comparaison des deux métriques de croisements

Nous avons vu plus tôt qu'il devient difficile d'établir des règles claires en se restreignant à la visualisation de bonnes solutions. Nous allons essayer de déterminer quel croisement est le plus intéressant à étudier, en utilisant une visualisation plus précise.

4.1.2 Étude visuelle des caractéristiques

Nous cherchons ici à déterminer s'il existe un lien entre certaines caractéristiques d'une solution, et la qualité de leur solution. Une méthode simple consiste à projeter les solutions sur un espace constitué uniquement des caractéristiques à étudier.

Application à l'étude des croisements

Dans la figure 4.5, les solutions sont projetées sur un espace en deux dimensions, représentant respectivement le nombre de croisements sur le chemin réel et le nombre de croisements sur le chemin euclidien. Enfin, pour visualiser la qualité des solutions, un code couleur a été ajouté. Chaque couleur correspond à une qualité de solution :

- pour les solutions quasi-optimales : écart à la meilleure solution connue $\leq 1\%$
- pour les solutions dont l'écart est compris entre 1% et 2%
- pour les solutions dont l'écart est compris entre 2% et 3%
- pour les solutions dont l'écart est compris entre 3% et 4%
- pour les solutions dont l'écart est compris entre 4% et 5%

Chaque point représente donc le nombre de croisements de chaque type, pour une solution. Plusieurs solutions y sont affichées, provenant de plusieurs instances similaires à celle représentée par la figure 4.1.

On observant la première dimension, il est possible de remarquer que pour n'importe quelle quantité de croisements réels, il existe toujours plusieurs solutions de très bonne qualité, ainsi que de nombreuses solutions de très mauvaise qualité. Il est donc difficile d'établir une règle liant la qualité d'une solution et le nombre de croisements réels.

Cependant, en étudiant le nombre de croisements euclidiens, nous pouvons remarquer que la plupart des solutions de très bonne qualité ont moins de 30 croisements euclidiens.

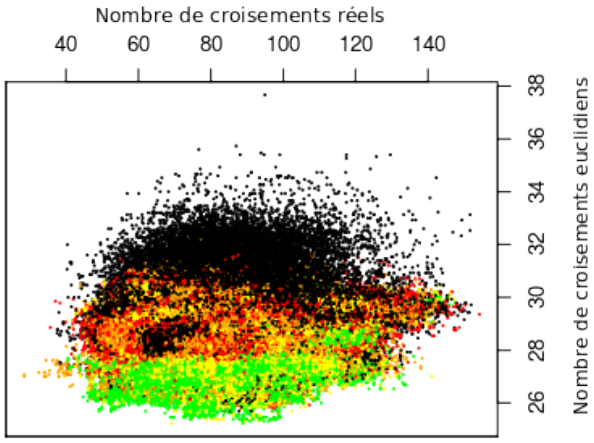


FIGURE 4.5 – Comparaison des fréquences de chaque type de croisement

Une simple projection en deux dimensions a donc permis d'établir une règle sur le nombre de croisements euclidiens, qui semble marcher sur plusieurs instances.

Vers une généralisation de la méthodologie ?

Il est donc tentant de se tourner uniquement vers cette méthode. Cependant, il est fréquent qu'une projection sur un espace à deux dimensions n'apporte aucune nouvelle connaissance. Le graphe 4.6 montre un cas de projection où les solutions de bonne et mauvaise qualité semblent réparties de manière chaotique.

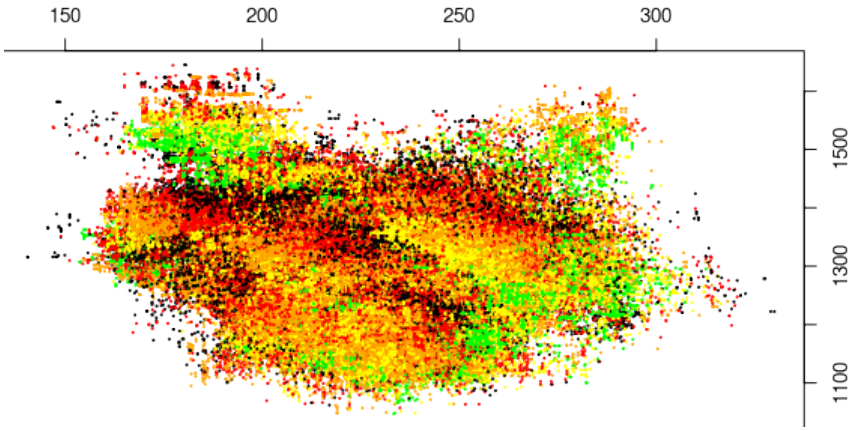


FIGURE 4.6 – Exemple de répartition chaotique

Le nombre de projections possibles augmente également très vite. Si on augmente le nombre de critères d'études, le nombre de projection devient vite difficile à étudier à la

main. La figure 4.7, montre les 45 projections obtenues lors de l'étude de 10 critères, qui seront expliqués plus tard dans ce chapitre. C'est encore réalisable manuellement, et il est même possible d'observer rapidement que les bonnes solutions sont souvent associées à une valeur élevée de la caractéristique S9, ou au contraire, à une valeur faible de S10, sans avoir à connaître la signification de ces critères.

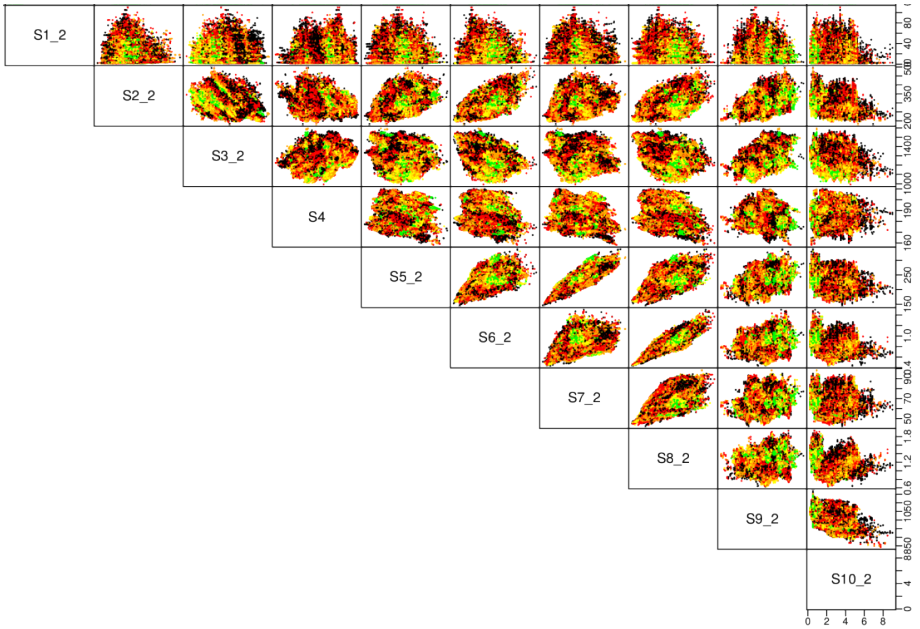


FIGURE 4.7 – Visualisation de 10 caractéristiques supplémentaires

Cependant, le nombre de caractéristiques peut encore augmenter. Reprenons nos instances simulant des distances non euclidiennes. Cette fois-ci, nous allons diversifier les catégories de véhicules utilisés, en incluant :

- un véhicule léger non motorisé, avec une vitesse maximale réduite, ne subissant pas de pénalité et une capacité de stockage très faible,
- un véhicule léger motorisé, avec une vitesse maximale moins faible, subissant une légère pénalité en zone dense et avec un stockage toujours faible,
- un véhicule de taille standard, avec une vitesse importante, et une grande capacité de stockage. Ce véhicule est fortement pénalisé en zone dense,
- un véhicule imposant, avec une très grande capacité de stockage, une vitesse réduite, et l'impossibilité d'accéder aux zones à forte densité.

Comme illustré dans la figure 4.8, chaque catégorie de véhicule a ses propres règles, ce qui signifie dans notre exemple qu'il faudra diviser chaque catégorie en 4 sous-catégories.

En supposant que l'on étudiait 10 critères, le passage des instances à une catégorie de véhicules, aux instances à 4 catégories font exploser le nombre de projections possible de 45 à 435. Il est donc impossible en pratique d'étudier rigoureusement l'ensemble des projections.

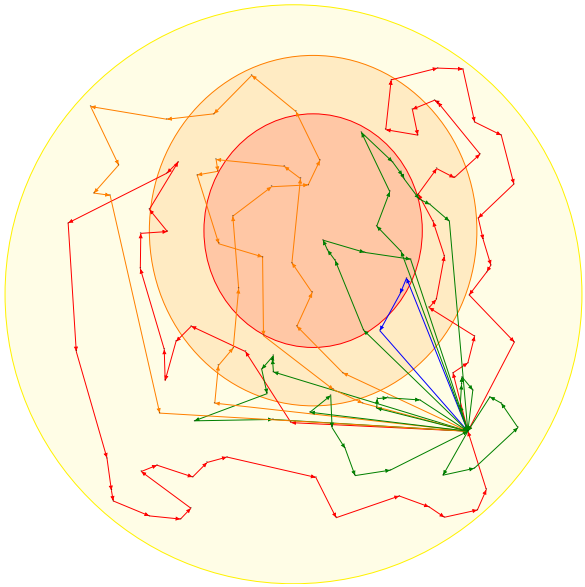


FIGURE 4.8 – Exemple d'instance hétérogène

Une projection plus simple

Si les projections par paires de caractéristiques ne marchent pas, il est toujours possible d'étudier les projections sur chaque caractéristique, individuellement. Ces règles établies permettront de comprendre la valeur optimale pour chacun des critères, et donc d'adapter le solveur pour qu'il intègre ces règles simples.

En étudiant les projections de la figure 4.9, il est très tentant d'établir des liens entre les caractéristiques et la qualité des solutions. Certaines semblent devoir être minimisées (S1, S10), d'autres au contraire doivent être maximisées (S8, S9), et certaines doivent avoir une valeur intermédiaire (S2, S7).

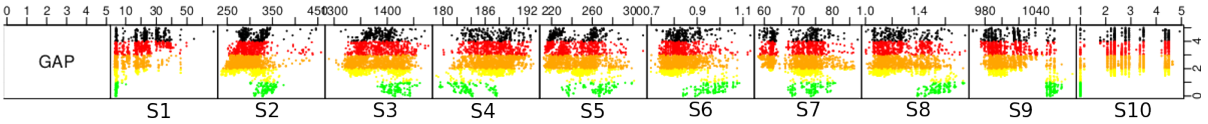


FIGURE 4.9 – Exemples de projections à une dimension

Cependant, toutes les solutions affichées sur la figure 4.9 proviennent de la même instance. En combinant les solutions de plusieurs instances, on obtient la figure 4.10. Il n'est désormais plus possible d'établir le moindre lien entre les caractéristiques et la qualité des solutions.

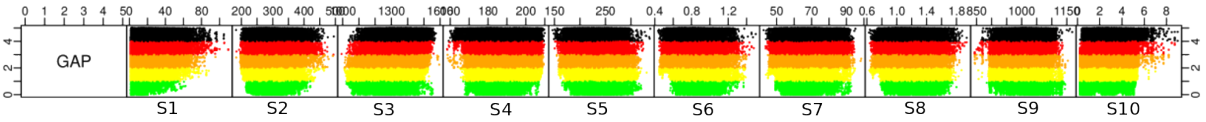


FIGURE 4.10 – Exemples de projections à une dimension, provenant de 81 instances

Conclusion Bien que les méthodes intuitives de visualisation soient simples à comprendre, et à mettre en œuvre, celles-ci sont assez vite limitées. Lorsque les problèmes à traiter gagnent en complexité, ces méthodes n'apportent que des connaissances partielles et nécessitent une expertise sur les données à étudier. De plus, les conclusions obtenues sur une instance ne sont pas toujours généralisables. Il convient donc de se tourner vers des méthodes plus rigoureuses, automatisables d'extraction de connaissances.

4.2 Apports de la fouille de données

La fouille de données consiste à extraire des informations d'un ensemble de données brutes. Par convention, ces données sont présentées sous la forme d'un tableau, tel le tableau 4.1. Chaque ligne correspond aux individus de la population d'étude. Dans le contexte des tournées de véhicules, les lignes correspondront à des solutions, ou à des instances. En colonne sont représentés les attributs des individus étudiés.

Le but de la fouille de données est d'extraire des connaissances sur les liens entre les caractéristiques des individus. Ces règles peuvent permettre d'identifier les individus similaires via des algorithmes de partitionnement. La fouille de données permet également de construire des hypothèses à partir d'un ensemble de caractéristiques, pour prédire la valeur d'une autre caractéristique.

Dans l'exemple du tableau 4.1, les lignes représentent des solutions. En colonnes, il y p caractéristiques, notées $F_1, \ldots, F_p$, ainsi qu'une caractéristique supplémentaire, appelée *Qualité*. La fouille de données peut permettre d'identifier des relations entre les caractéristiques $F_1, \ldots, F_p$ et la caractéristique *Qualité*. Un exemple de règle peut alors être :

Si $F_1 > 1.5$ et $F_2 < 0.3$, alors Qualité $= 0$.

	Qualité	F_1	F_2	...	F_p
n solutions	1	1.4	0.6	...	2.9
	0	1.6	0.2	...	1.7
	1	1.2	1.8	...	2.0
	⋮	⋮	⋮	⋮	⋮
	1	1.4	0.7	...	2.5
		p caractéristiques			

Tableau 4.1 – Exemple de jeu de données

4.3 Caractéristiques issues de la littérature

Dans le chapitre 2, nous avons vu que plusieurs articles (Kanda *et al.*, 2016; Arnold et Sörensen, 2019b) évoquent des caractéristiques permettant d'étudier des solutions du TSP ou du CVRP sans les expliciter. Ici, nous nous concentrerons uniquement sur celles créées par Arnold et Sörensen (2019b), qui ont été longuement étudiées pendant la thèse.

Dans leur article, K. Sörensen et F. Arnold ont essayé d'identifier les caractéristiques permettant de reconnaître une solution quasi optimale, ou au contraire, une solution non optimale. Celles-ci sont énumérées dans le tableau 4.2.

S1	Nombre moyen de croisements, par client ;
S2	Plus grande distance entre deux clients consécutifs, par tournée ;
S3	Distance moyenne entre les clients de début/fin de tournée, et le dépôt ;
S4	Distance moyenne entre les centre de gravité des tournées ;
S5	Largeur moyenne des tournées ;
S6	Envergure moyenne des tournées ;
S7	Compacité moyenne des tournées, en largeur ;
S8	Compacité moyenne des tournées, en envergure ;
S9	Profondeur moyenne des tournées ;
S10	Écart type du nombre de clients de chaque tournée.

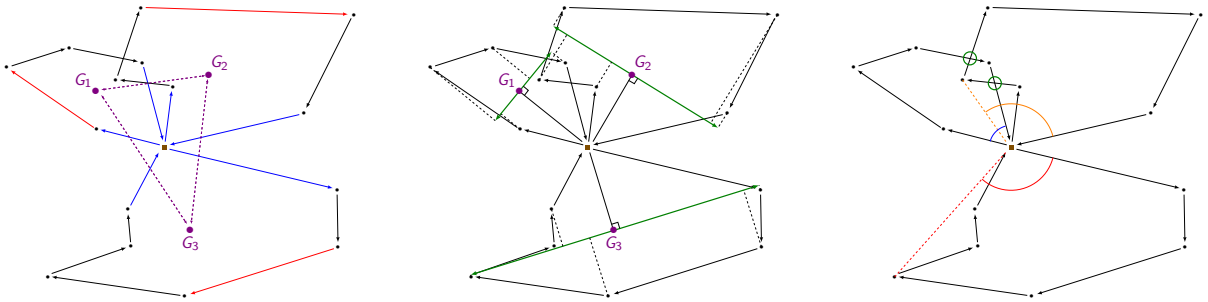
Tableau 4.2 – Caractéristiques issues des solutions

Cependant, deux solutions aux caractéristiques similaires, mais issues de deux instances différentes sont potentiellement de qualité différente. Afin de vérifier cette hypothèse, les auteurs ont testé 8 nouvelles caractéristiques issues des instances. Ces caractéristiques sont précisées dans le tableau 4.3.

I1	Nombre de clients ;
I2	Nombre de véhicules ;
I3	Degré de capacité d'utilisation ;
I4	Distance moyenne entre les clients, pair à pair ;
I5	Écart type de la distance entre les clients ;
I6	Distance moyenne des clients vers le dépôt ;
I7	Écart type de la distance des clients vers le dépôt ;
I8	Écart type sur l'angle entre le dépôt et les pairs de clients ;

Tableau 4.3 – Caractéristiques issues des instances

Ces caractéristiques sont essentiellement visuelles et intuitives. En effet, il semble pertinent d'éviter de trop croiser les tournées, ou de minimiser la taille des arêtes entre deux clients consécutifs. La figure 4.11 permet de visualiser ces caractéristiques via une solution utilisant trois véhicules.



(a) S2, S3, S4 (b) S5 et S7 (c) S1, S6 et S8

FIGURE 4.11 – Explication des caractéristiques

— Il y a deux croisements (encerclés en vert dans le graphe 4.11(c)) et 15 clients. On a donc $S1 = 2/15$.

— Les arêtes rouges dans la figure 4.11(a) représentent la plus longue distance entre deux clients consécutifs, pour chaque tournée. La caractéristique S2 est donc égale à la taille moyenne de ces arêtes.

— S3 est égale à la taille moyenne des arêtes bleues de la figure 4.11(a), représentant les arêtes connectées au dépôt, pour chaque tournée.

— Les points violets identifiés par G_i représentent les centres de gravité de chaque tournée. La caractéristique S4, représente la distance moyenne entre ces points, représentée par les arêtes violettes.

— S5 est égale à la taille moyenne des arêtes vertes de la figure 4.11(b). Celles-ci correspondent aux distances maximales de deux clients d'une même tournée, sur la projection sur l'axe perpendiculaire à l'axe dépôt-centre de gravité.

— La caractéristique S7 correspond à la moyenne de ces distances projetées.

— S6 est égale à la valeur moyenne des angles maximums obtenus entre deux clients d'une tournée, et le dépôt, représentés en 4.11(c).

— La caractéristique S8 correspond à la valeur moyenne des angles possibles entre deux clients d'une même tournée et le dépôt.

— La caractéristique S9 correspond à la distance moyenne par tournée entre le client le plus éloigné du dépôt et le dépôt. Dans notre exemple, il y a 15 clients et 3 tournées, de respectivement 4, 5 et 6 clients. Il y a donc en moyenne 5 clients par tournées.

— On a donc $S10 = \sqrt{\frac{(4-5)^2+(5-5)^2+(6-5)^2}{3}} \sim 0.82$

Plus de détails sur ces caractéristiques sont accessibles en lisant l'article original de Arnold et Sörensen (2019b).

4.4 L'apport explicatif de l'apprentissage automatique

Arnold et Sörensen (2019b) ont utilisé les 18 caractéristiques détaillées ci-dessus afin de prédire la qualité (proche de l'optimal, ou non-optimale) d'une solution. Les solutions utilisées viennent de 8 jeux d'instances, disponibles sur le lien suivant : `http://antor.uantwerpen.be/problem-knowledge/`. Par souci de cohérence, les expérimentations de cette section, et les figures 4.16, à 4.20 ont été réalisées d'après le même jeu de données "*gap2_large_center_highVariance.csv*". Cependant, les éventuelles différences entre les jeux de données seront détaillées quand cela sera nécessaire.

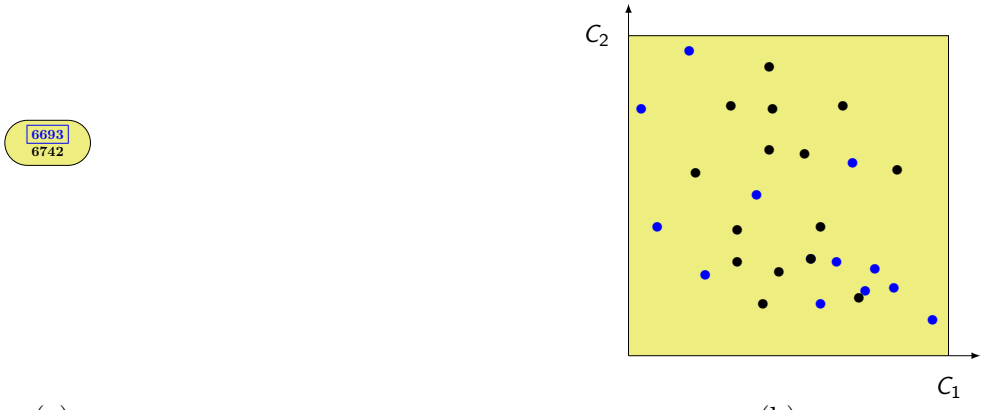
4.4.1 Comment distinguer les caractéristiques indispensables ?

Une premières méthode visuelle : l'arbre de décision

L'arbre de décision est une méthode efficace pour comprendre l'impact de chaque caractéristique sur la valeur d'une variable cible. En partant de l'espace entier des caractéristiques, l'arbre de décision va déterminer la caractéristique la plus discriminante, ainsi qu'une valeur seuil.

Dans les figures 4.12 à 4.15, nous étudierons un cas fictif où seulement deux caractéristiques sont suffisantes pour décrire la qualité d'une solution. L'arbre est représenté par les figures 4.12(a) à 4.15(a), tandis que les figures 4.12(b) à 4.15(b) représentent les zones de l'espace des caractéristiques créées par l'arbre.

Nous cherchons à établir des règles, en fonction des caractéristiques C_1 et C_2 permettant de déduire la qualité d'une solution. Dans notre base d'apprentissage, nous disposons de 6693 solutions de bonne qualité, représentées par les points bleus dans la figure 4.12(b), et de 6742 solutions de mauvaise qualité, représentées par les points noirs.



(a) (b)

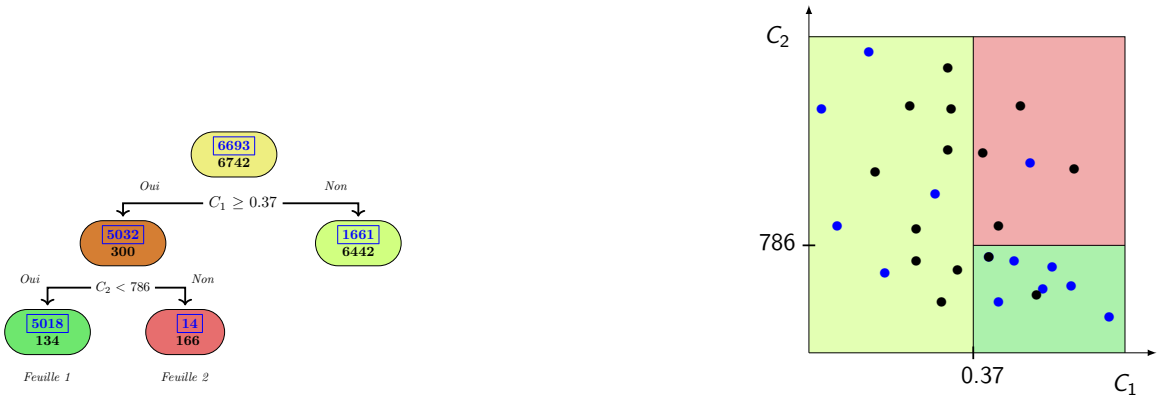
FIGURE 4.12

Il faut désormais déterminer un hyperplan séparant le domaine des caractéristiques en deux sous-domaines. Cet hyperplan ne peut être défini que par une seule caractéristique. Dans notre espace à deux dimensions, il ne peut donc y avoir que des coupes horizontales ou verticales. Ici, l'hyperplan défini par $C_1 = 0.37$ est celui amenant a une plus grande homogénéité des sous espaces séparés. On obtient ainsi un premier sous-espace, défini par $C_1 \geq 0.37$, contenant 5032 solutions de bonne qualité et 300 solutions de mauvaise qualité. Le deuxième sous-espace contient 1661 solutions de bonne qualité et 6442 solutions de mauvaise qualité.

(a) (b)

FIGURE 4.13

Cette étape est ensuite répétée de façon récursive à chacun des nœuds. Dans la figure 4.14, l'arbre s'occupe de diviser l'hyperplan $C_1 \geq 0.37$. Cette fois, la caractéristique C_2 est la plus discriminante. On obtient ainsi deux nouveaux sous-espaces. Un premier ensemble est défini par $C_1 \geq 0.37$ et $C_2 < 786$, contenant 5018 solutions de bonne qualité et 134 solutions de mauvaise qualité. Le deuxième ensemble est défini par $C_1 \geq 0.37$ et $C_2 \geq 786$, et contiennent 14 solutions de bonne qualité et 166 solutions de mauvaise qualité.

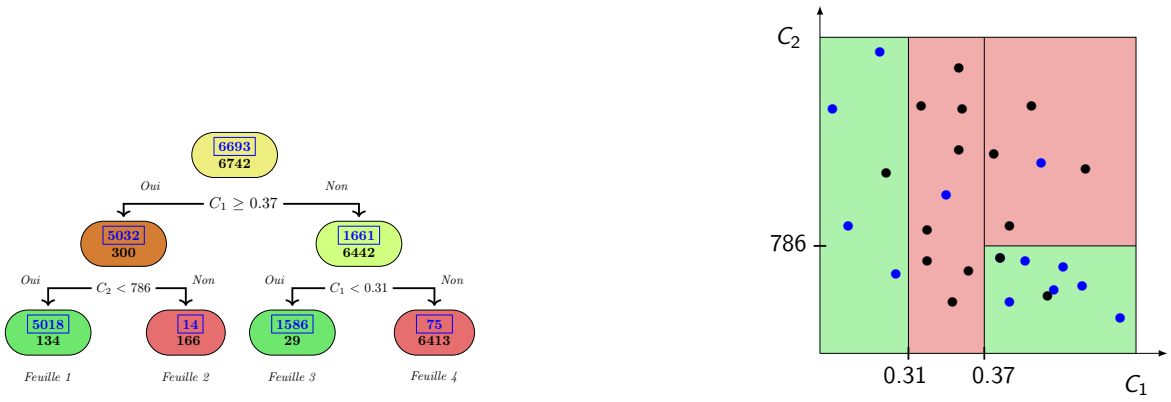


(a) (b)

FIGURE 4.14

Partitionner l'un ou l'autre de ces ensembles ne va pas davantage augmenter leur homogénéité, qui est déjà importante. L'arbre passe donc à l'étude de la zone $C_1 < 0.37$.

Cette zone peut être séparée en utilisant à nouveau la caractéristique C_1. Les nouvelles zones, définies respectivement par $C_1 < 0.31$ et $C_1 \in [0.31; 0.37[$ sont principalement constituées d'une seule catégorie de solutions.



(a) (b)

FIGURE 4.15

À la fin de l'algorithme, l'espace des caractéristiques est donc divisé en 4 zones :

— 2 zones contenant majoritairement des solutions de bonne qualité, représentées par les feuilles 1 et 3.

— 2 zones contenant majoritairement des solutions de mauvaise qualité, représentées par les feuilles 2 et 4.

L'arbre de décision est donc un outil puissant pour définir des règles de décisions simples. Ces règles permettent de mieux comprendre ce qui fait une bonne solution.

Nous pouvons désormais appliquer cette méthode sur le jeu de données de référence *gap2_large_center_highVariance.csv*. L'arbre ainsi obtenu est représenté par la figure 4.16. Ici, chaque feuille contient 3 informations :

- La catégorie majoritaire :
  - 0 : La majorité des solutions sont non-optimales
  - 1 : La majorité des solutions sont quasi-optimales
- La proportion de solutions non-optimales
- La proportion de solutions optimales

Par exemple, la première feuille de la figure 4.16 contient une majorité de solutions non optimales. Plus précisément, elle contient 80% de solutions non-optimales et 20% de solutions quasi-optimales.

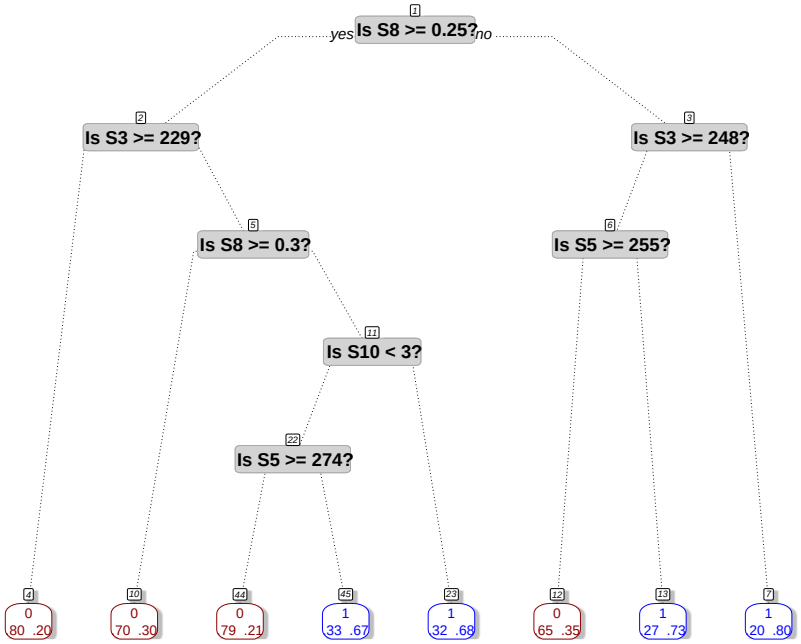


FIGURE 4.16 – Exemple de règles définissant des solutions de bonne qualité

Les arbres de décision nous permettent de comprendre les liens entre les caractéristiques et la qualité des solutions. Par exemple, sur la figure 4.16, les trois feuilles sur la droite, avec une valeur faible pour la caractéristique S8 contiennent plus de solutions quasi-optimales que les feuilles avec une valeur élevée de la caractéristique S8. Cette caractéristique correspond à la compacité moyenne des tournées (plus cette valeur est faible, plus les tournées sont compactes). L'arbre semble indiquer que les tournées doivent être aussi compactes que possible. La même conclusion peut-être obtenue avec les caractéristiques S3 (distance moyenne entre le dépôt, et les premiers/derniers clients des tournées) et S5 (largeur moyenne des tournées).

L'arbre de décision créé par F. Arnold et K. Sörensen sépare la caractéristique S8 autour de la valeur 0.25, et la caractéristique S3 autour des valeurs 167, 199, 222, 240, 294. L'arbre que nous avons implémenté sépare la caractéristique S8 en 0.25, et la caractéristique S3 en 229 et 248. Les caractéristiques choisies et les valeurs de séparations étant similaires, nous pouvons confirmer les travaux de F. Arnold et K. Sörensen.

Métriques de l'utilité d'une solution

Il est possible d'aller plus loin dans l'interprétation avec des méthodes ensemblistes, comme la forêt aléatoire. Cette méthode obtient de meilleurs résultats en combinant des milliers d'arbres de décision. Les forêts aléatoires permettent ainsi d'étudier en détail l'intérêt de chaque caractéristique pour chaque arbre de décision.

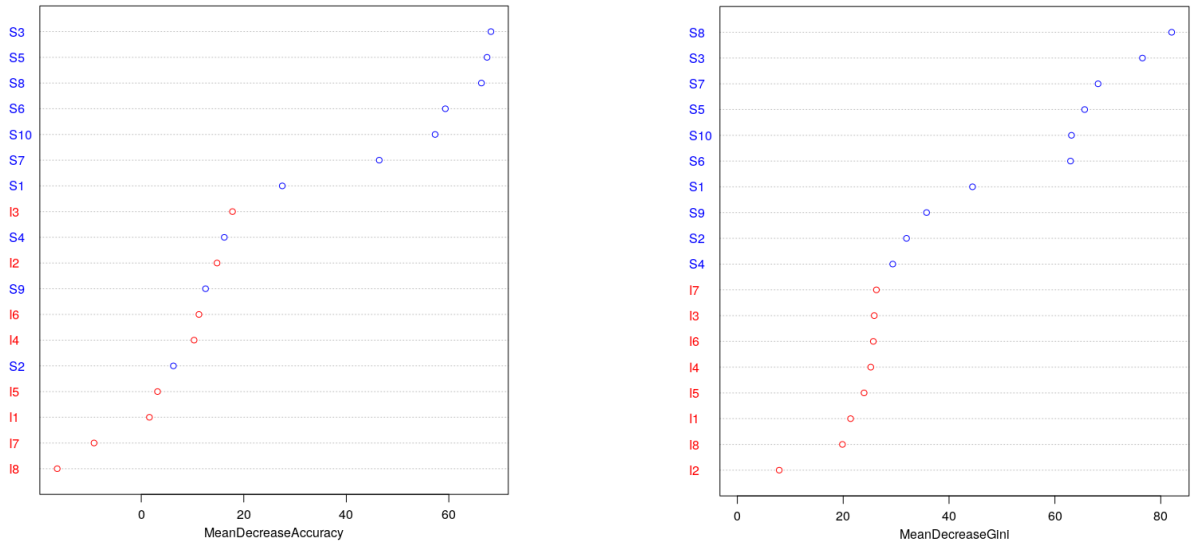


FIGURE 4.17 – Indice d'utilité des différentes caractéristiques (tri croissant)

La figure 4.17 détaille une partie de ces analyses statistiques. On y voit les valeurs de *MeanDecreaseAccuracy* à gauche qui indique la perte d'information moyenne lorsque la caractéristique étudiée est remplacée aléatoirement par une autre. Le graphe *MeanDecreaseGini* à droite représente la réduction moyenne du coefficient de Gini (Breiman *et al.*, 1984) lors de la séparation d'un nœud. Le coefficient de Gini représente l'hétérogénéité d'un nœud. Plus ce coefficient est élevé, plus un nœud est de qualité hétérogène. Inversement, un coefficient de Gini de 0 indique que le nœud contient des solutions de qualité identique.

En résumé, la figure 4.17 trie les caractéristiques par ordre d'importance selon deux critères différents.

Comme énoncé plus tôt, deux catégories de caractéristiques sont étudiées, celles tirées des solutions (S), et celles provenant des instances (I). Les premières sont affichées en bleu sur la figure 4.17, tandis que les secondes sont affichées en rouge. Cette figure semble donc indiquer que les caractéristiques $S1$ à $S10$ (provenant des solutions) sont plus importantes que les caractéristiques $I1$ à $I8$ (provenant des instances). Cette idée a été confirmée

dans nos expérimentations. On peut observer par les variables I1 à I8 que les valeurs de *meanDecreaseAccuracy* et *meanDecreaseGini* sont respectivement inférieures à 20 et 30. Les caractéristiques S3, S5, S6, S8 et S10 ont des valeurs supérieures à 60 pour ces métriques.

Afin de confirmer nos conclusions, nous avons fait une méta-analyse sur les autres jeux de données fournis par les auteurs. Nous avons considéré qu'une caractéristique est significative si sa valeur de *meanDecreaseAccuracy* et *meanDecreaseGini* est supérieure ou égale à 60. Les caractéristiques S1, S3, S5, S6, S7 et S8 sont considérées significatives pour au moins 6 des 16 jeux de données. Les autres caractéristiques ne sont pas considérées comme significatives dans au moins 14 jeux de données.

Expérimentations

Pour confirmer que les caractéristiques I1 à I8 ne sont pas nécessaires, nous avons effectué de nouvelles expérimentations. Pour cela, nous avons testé l'efficacité des deux méthodes de classification du papier original, la forêt aléatoire, et le *Support Vector Machine* (SVM). Ces méthodes ont été testées une première fois avec l'ensemble des caractéristiques, puis une deuxième fois, sans les caractéristiques $I1$ à $I8$.

Pendant une expérimentation en apprentissage automatique, il est nécessaire d'utiliser une partie des données pour générer les règles de décision. Cette partie est appelée *ensemble d'apprentissage*, ou encore *training set*. Les données restantes servent à vérifier l'efficacité du modèle précédemment créé. Ce second ensemble est appelé *ensemble de test* ou *test set*.

Pendant nos expérimentations, nous avons testé 50 fois chaque paire algorithme / caractéristique, en faisant varier les ensembles de test et d'apprentissage. Ces ensembles, ont été définis aléatoirement, en gardant une proportion classique pour ce genre d'expérimentation : 2/3 pour l'apprentissage, et 1/3 pour le test.

Enfin, nous étudions la proportion de solutions de l'ensemble de test dont la qualité (quasi optimale ou non optimale) a été correctement prédite. Les performances moyennes sont résumées dans le tableau 4.4.

Caractéristiques	Forêt aléatoire	Support Vector Machine
$I1 \dots I8, S1 \dots S10$	75.16%	77.28%
$S1 \dots S10$	76.10%	77.16%

Tableau 4.4 – Expérimentations sur l'utilité des caractéristiques I1 à I8

On peut observer des performances similaires, même après la suppression de 8 caractéristiques sur 18. On peut donc bel et bien conclure de l'inutilité des caractéristiques liées aux instances, au moins pour le jeu de données "*gap2_large_center_highVariance.csv*".

Pour la suite de l'étude, nous n'utiliserons plus les caractéristiques I1 à I8.

4.4.2 Réduction de l'espace à étudier grâce à l'ACP

Étape préliminaire : l'analyse des corrélations

La figure 4.18 montre le coefficient de corrélation de Pearson entre toutes les paires de caractéristiques. Une valeur proche de 1 indique une très forte corrélation entre les variables (les deux variables ont des valeurs faibles/élevées sur les mêmes jeux de données). Une valeur proche de -1 indique une anti-corrélation (la variable 1 est élevée quand la variable 2 est faible, et inversement). Enfin, une valeur proche de 0 indique une absence de corrélation entre les deux caractéristiques.

Comme indiqué dans la légende, la couleur de chaque disque indique le niveau de corrélation. Plus le rayon du disque est élevé, et la couleur intense, plus la corrélation (ou anti-corrélation) est importante.

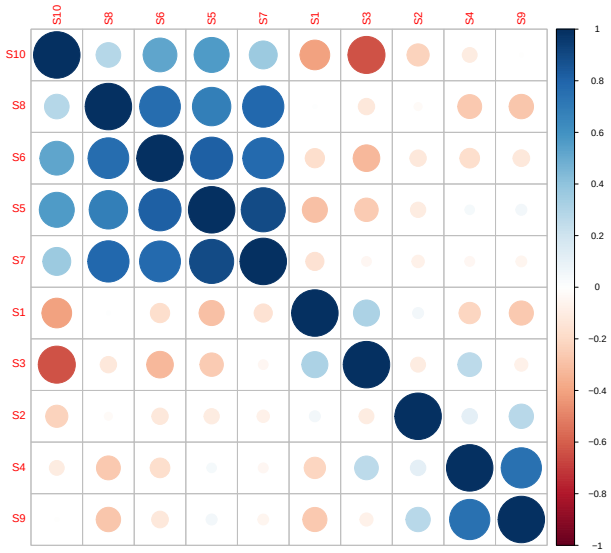


FIGURE 4.18 – Matrice de corrélation des caractéristiques des solutions

Avec une valeur de corrélation supérieure à 0.7, les caractéristiques S5, S6, S7 et S8 sont très fortement corrélées deux à deux. De manière similaire, on peut observer la corrélation entre les caractéristiques S4 et S9. Inversement, avec une corrélation inférieure à -0.7, les caractéristiques S3 et S10 sont fortement anti-corrélées.

Cette simple description met en lumière les liens et redondances entre les variables.

Ces liens ont également été observés dans les autres jeux de données. Plus précisément, la corrélation entre les caractéristiques S5, S6, S7, S8 a été observée dans tous les jeux de données. Les caractéristiques S4 et S9 sont fortement corrélées dans 9 jeux de données, et non corrélées autrement. Les caractéristiques S3 et S10 sont toujours plus ou moins anti-corrélées, avec une corrélation toujours inférieure à -0.4. Enfin, les caractéristiques S3 et S6 sont fortement anti-corrélées sur 11 des 18 jeux d'instances. Une tendance semble donc se profiler sur les liens entre les caractéristiques.

Analyse en Composantes Principales : principe

L'*Analyse en Composantes Principales* (ACP) projette les variables sur un sous-espace de dimension réduite permettant une meilleure interprétation de la structure des données. Plus précisément, les composantes principales sont des combinaisons linéaires des caractéristiques S1 à S10. Les figures 4.19 et 4.20 montrent les projections selon deux des trois directions les plus représentatives. Une direction est définie comme représentative si elle explique une grande partie de la variabilité des données, c'est-à-dire l'inertie du nuage de points.

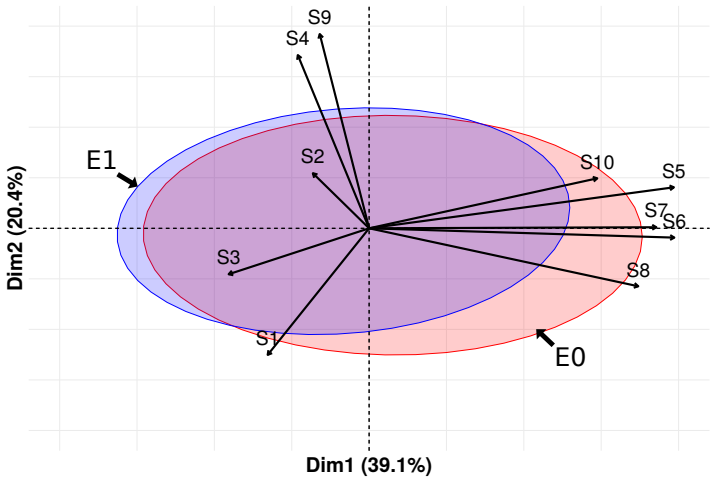


FIGURE 4.19 – ACP selon les deux premières composantes

Sur ces figures, les vecteurs représentent la projection des variables initiales sur le sous-espace. Plus un vecteur est long, plus la caractéristique associée est importante dans ce sous-espace. L'angle entre deux vecteurs indique la corrélation entre les variables associées. L'ellipse bleue (E1) englobe 95% des solutions quasi-optimales, tandis que l'ellipse rouge (E0) englobe 95% des solutions non optimales.

Résultats de l'ACP

La figure 4.19 donne la projection sur les deux dimensions les plus significatives. Celles-ci expliquent 59% de la variance des données (respectivement, 39% et 20% pour les dimensions 1 et 2). En étudiant la longueur des vecteurs, on peut en déduire que les caractéristiques S6, S5 et S7 sont bien représentées, contrairement à S2 qui n'est que peu représentée.

L'angle entre les caractéristiques S6 et S7 est presque nul, indiquant leur forte corrélation. Inversement, l'angle entre les caractéristiques S1 et S8 est proche de $\frac{\pi}{2}$, indiquant leur absence de corrélation. Enfin, l'angle entre les caractéristiques S3 et S10 est proche de π et révèle leur anti-corrélation.

On peut également observer que les ellipses indiquant respectivement, l'emplacement de 95% des solutions quasi-optimales, ou non-optimales, ont une forte intersection. Il est donc impossible, avec une projection sur une combinaison linéaire en deux dimensions, de classer la majorité des solutions.

En observant quelles caractéristiques sont les plus proches de chaque dimension, il est possible d'interpréter celles-ci. On ne considérera ici que les caractéristiques ayant plus de 0.7 en valeur de corrélation. En étudiant la valeur absolue de chaque vecteur, une fois projeté sur la première dimension, on peut en déduire que cette dernière est majoritairement représentée, par ordre décroissant d'importance, par les caractéristiques S6, S5, S7 et S8. Celles-ci représentent toutes une façon différente de mesurer la compacité moyenne des tournées. Par conséquent, la première dimension représente la compacité des tournées. De façon équivalente, la deuxième dimension est représentée par les caractéristiques S4 et S9. Ainsi, celle-ci représente la profondeur et l'équilibre géographique des tournées.

Prendre en compte la troisième dimension permet de gagner en compréhension. Cette dernière représente les distances entre les premiers/derniers clients des tournées, et le dépôt.

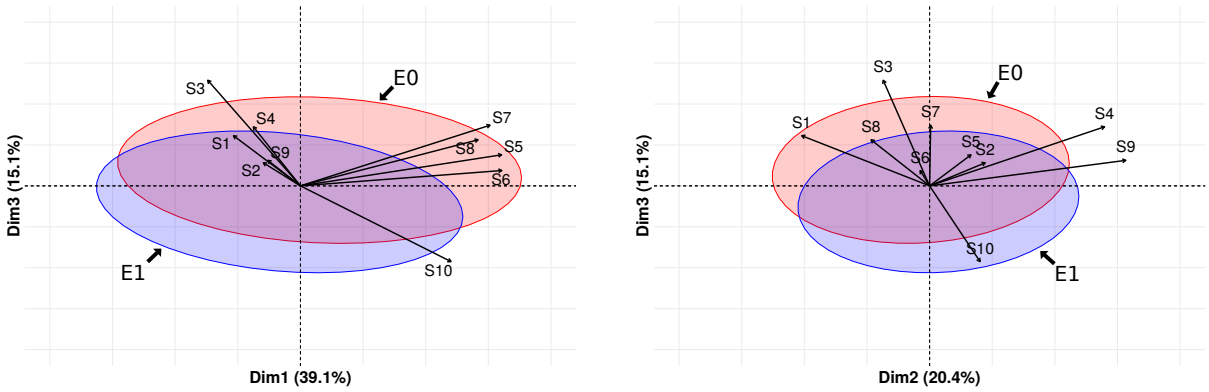


FIGURE 4.20

La figure 4.20 précise un peu plus la différence entre les solutions quasi-optimales, et les solutions non optimales. L'intersection entre les deux ellipses est moins importante selon cette troisième dimension.

En étudiant la taille du vecteur associée à la caractéristique 2 dans la figure 4.19, on pouvait remarquer que celle-ci ne semblait pas importante. Cette observation semble se confirmer avec la figure 4.20. La taille de la plus grande arête entre deux clients consécutifs n'est donc peut-être pas aussi intéressante qu'on pouvait le supposer.

Quelle que soit la dimension d'observation, l'angle entre les vecteurs des caractéristiques S5 à S8 est toujours petit, montrant leurs corrélations. Cela paraît logique, étant donné que ces caractéristiques représentent quatre façons de calculer la compacité moyenne des tournées. Ces caractéristiques sont principalement associées à des solutions de mauvaise qualité (leurs vecteurs pointent vers des zones couvertes uniquement par l'ellipse E0). Ces caractéristiques sont donc à minimiser.

4.5 Conclusion

Nous avons vu, au début de ce chapitre, que l'analyse visuelle de solutions, ainsi que de leurs caractéristiques permet d'obtenir une première idée de la composition d'une bonne solution. Cependant, ces méthodes sont limitées car elles ne sont pas toujours généralisables à l'ensemble des instances existantes. Ces travaux ont donc montré la nécessité d'utiliser des méthodes automatisables pour l'extraction de connaissance.

Parmi ces méthodes, l'analyse factorielle nous aide à améliorer le processus de choix des variables. Elle permet d'ouvrir une nouvelle voix sur l'amélioration de la précision des méthodes d'apprentissage automatique.

En voulant valider notre hypothèse sur l'inutilité des caractéristiques I1 à I8, plusieurs expérimentations ont été effectuées. Pour chaque test, 50 forêts aléatoires contenant exactement 2000 arbres ont été implémentés. Lorsqu'ils sont testés avec la totalité des caractéristiques, nos forêts aléatoires réussissent à identifier correctement la qualité des solutions dans 75.16% des cas. En enlevant les caractéristiques I1 à I8, l'efficacité monte à 76.1%. De plus, en enlevant également la caractéristique S2, on obtient une fiabilité de 76.32%. Par conséquent, supprimer une caractéristique a priori inutile a permis d'améliorer très légèrement la performance de la forêt aléatoire.

Ainsi, nous avons prouvé l'inefficacité des caractéristiques $I1$ à $I8$ et S2 sur les jeux de donnés fournis par K. Sörensen et F. Arnold. Enfin, ajouter des caractéristiques inutiles semble réduire l'impact de celles intéressantes, réduisant ainsi la pertinence finale de la prédiction.

Les travaux effectués au cours de ce chapitre ont permis d'apporter des éléments de réponse concernant la question de recherche 2 (*comment caractériser une solution d'un problème de tournée ?*) en mettant en place les outils nécessaires pour extraire des connaissances sur la caractérisation des bonnes solutions.

Chapitre 5

Méthode hybride fondée sur l'extraction de règles

Dans ce chapitre, nous allons voir comment les méthodes statistiques ou d'apprentissage vues au chapitre 4 peuvent être combinées au solveur du chapitre 3. Après une explication de l'idée générale, le solveur hybride sera présenté en détail. De nouvelles instances, issues de données réelles, seront introduites puis nous étudierons les performances du nouveau solveur sur ces instances. Enfin, cette section se conclura sur l'intérêt du solveur et les pistes d'amélioration.

Sommaire

5.1 Idée générale

Soit X l'ensemble des solutions d'un problème d'optimisation à minimiser. Notons $f : X \mapsto \mathbb{R}$ la fonction de coût associée à minimiser. Dans une méthode de recherche locale classique, une solution initiale x_0, de coût f_0 est améliorée jusqu'à atteindre un minimum local x_1, de coût $f_1 \leq f_0$.

Soit F l'espace représentant les valeurs des caractéristiques des solutions. Notons F^* l'estimation des caractéristiques de la solution optimale. Soit $s : F \mapsto \mathbb{R}$ une fonction de score, indiquant une estimation de la distance, dans F entre les caractéristiques de la solution courante et F^*.

Ainsi, lorsque le minimum local x_1 est atteint il est possible de continuer la recherche, en changeant la fonction de référence (voir figure 5.1). Le nouvel objectif est donc de minimiser le score de la solution, au lieu de son coût.

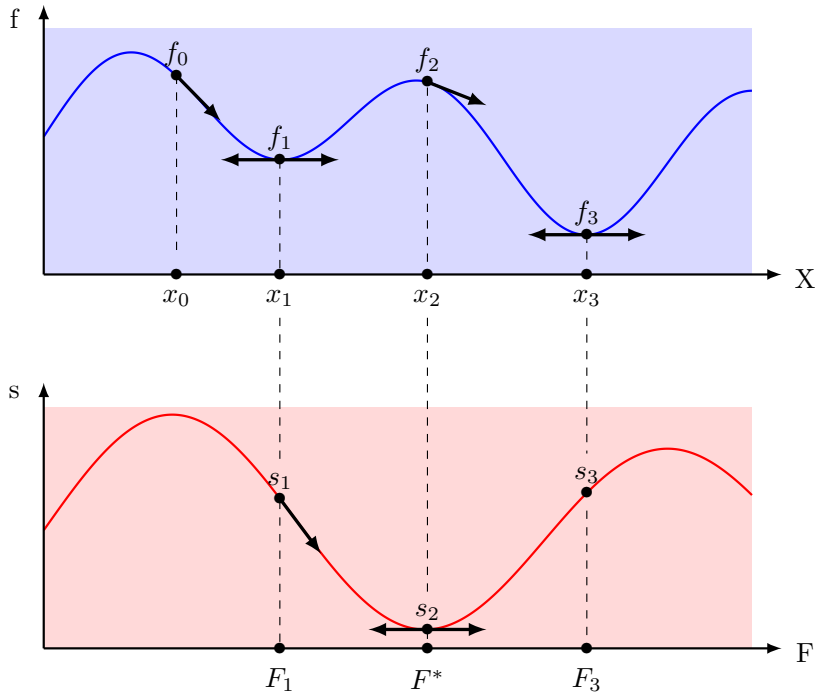


FIGURE 5.1 – Fonction de coût et score pour une même instance

Dans l'exemple de la figure 5.1, la recherche locale, basée sur le score et initialisée en x_1 mène à la solution x_2, dont le score s_2 est plus faible que le score initial s_1. Le coût f_2, est plus élevé que la solution obtenue précédemment, mais la solution x_2 est plus proche

dans X de la solution optimale que ne l'est x_1. Enfin, en effectuant une nouvelle recherche locale, basée sur la fonction de coût, la solution optimale x_3 est trouvée.

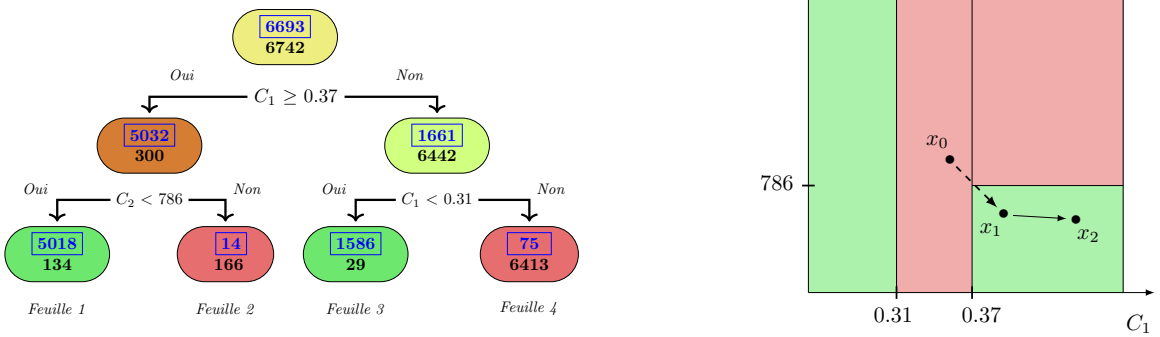
5.2 Description de la méthode hybride

La section précédente a montré qu'il est théoriquement possible de guider la recherche locale en changeant la fonction de recherche. Nous allons désormais voir comment fonctionne le solveur proposé.

5.2.1 Un solveur guidé par les caractéristiques

Nous avons vu dans le chapitre 4 qu'il est possible de partitionner l'espace des caractéristiques en sous-espaces selon la qualité des solutions associées.

Supposons que l'arbre de décision obtenu sur une instance soit celui de la figure 5.2(a), l'espace des caractéristiques est donc découpé en quatre sous-espaces, dont deux contenant majoritairement des solutions de bonne qualité, appelées *zones prometteuses*.



(a) (b)

FIGURE 5.2

L'idée est donc de faire une recherche locale cherchant à minimiser la distance entre la solution courante, et la zone prometteuse la plus proche.

Soit x_0, une solution dont les caractéristiques (C_1, C_2) sont $(0.36, 790)$, celles-ci sont proches de l'espace défini par $C_1 \geq 0.37$ et $C_2 < 786$. Après une recherche locale, il est donc possible d'obtenir la solution x_1, appartenant à la zone prometteuse. Cette solution n'est peut-être pas elle-même intéressante, mais elle se trouve dans une zone dense en

solutions de bonne qualité. Effectuer une recherche locale à partir de x_1 et basée sur le coût des solutions peut donc mener en quelques itérations à une solution x_2 de bonne qualité.

5.2.2 Organisation du solveur

Le solveur proposé, guidé par les caractéristiques, ou *Features Guided Multiple Neighborhood Search* (FG-MNS) est constitué de deux phases : l'*apprentissage* et l'*exploitation*.

Phase d'apprentissage

Pendant cette phase, le solveur détaillé dans le chapitre 3 est exécuté pour générer des solutions. Ensuite, un arbre de décision est lancé pour déterminer les zones prometteuses. Il existe plusieurs variantes d'apprentissages :

Apprentissage hors ligne L'arbre de décision est créé bien avant l'utilisation du solveur, en utilisant des solutions issues d'autres instances.

Apprentissage en ligne Dans cette variante, l'apprentissage se fait pendant l'exécution du FG-MNS, basé sur les solutions de l'instance à traiter.

Apprentissages hors ligne et en ligne Le solveur utilise dans un premier temps l'arbre de décision de l'apprentissage hors ligne pour guider la recherche locale. Au bout d'un temps t, défini en paramètre, un second arbre de décision est créé, basé sur les solutions obtenues lors de la première phase de résolution.

Phase d'exploitation

Dans cette phase, le solveur utilise les règles définies par l'arbre de décision pour guider la solution courante vers une zone prometteuse. Pour celui-ci, une recherche locale, basée sur la réallocation inter-route, cherchant à minimiser la distance dans l'espace des caractéristiques entre la solution et la zone prometteuse la plus proche. Une fois la zone prometteuse, ou un minimum local atteint, une nouvelle recherche est exécutée, basée sur le coût, en utilisant les voisinages décrits dans le chapitre 3.

Ces étapes sont ainsi répétées avec une nouvelle solution initiale, jusqu'à atteindre la limite de temps.

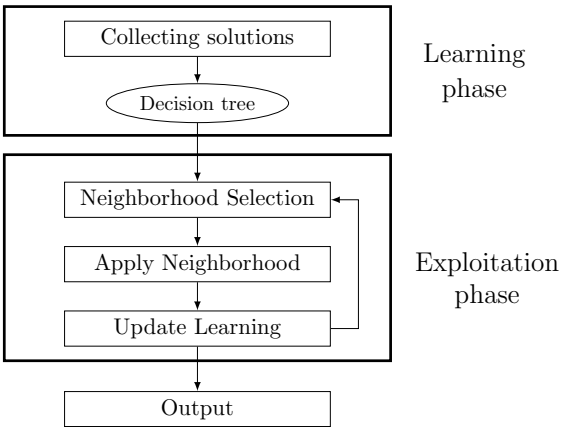


FIGURE 5.3 – Organigramme du FG-MNS

Dans le cadre d'un apprentissage à la fois hors ligne et en ligne, l'arbre de décision est mis à jour, en ne se basant que sur les nouvelles solutions rencontrées.

Une fois la limite de temps atteinte, l'algorithme s'arrête et retourne la meilleure solution obtenue.

5.2.3 Caractéristiques utilisées

Pour créer les arbres de décision, la plupart des caractéristiques suggérées par Arnold et Sörensen (2019b) ont été reprises. Cependant, certaines caractéristiques ont été modifiées et d'autres ont été ajoutées pour utiliser un maximum d'informations lors de la générations de règles. Enfin, les caractéristiques représentant la compacité moyenne des tournées ont été enlevée car elles sont redondantes avec les caractéristiques sur la largeur et l'envergure moyenne des tournées, comme indiquée dans l'étude du graphe de corrélation effectué au chapitre 4. Le tableau 5.1 montre l'ensemble des caractéristiques utilisées. Celles-ci sont principalement focalisées sur les tournées, qui sont elles-mêmes dépendantes de la catégorie des véhicules associées. Il existe donc une version de chaque caractéristique pour chaque catégorie de véhicules, à l'exception de la caractéristique S16 (distance moyenne entre les tournées) qui fait le lien entre les catégories de véhicules.

La plupart des nouvelles caractéristiques correspondent aux écarts-types sur des informations dont les moyennes sont déjà calculées. Cette information supplémentaire permet de connaître le degré de similarité entre les tournées. L'impact de la demande des clients les plus proches et les plus éloignés du dépôt est aussi étudié, ainsi que la proportion de certaines parties de la tournée sur l'ensemble du chemin parcouru. Enfin, le degré de voi-

sinage moyen des clients est étudié. Il s'agit de calculer la proximité moyenne des clients connectés entre eux : un client servi après son troisième voisin et avant son deuxième voisin a un degré de voisinage de 2.5.

S1	Largeur moyenne des tournées.
S2	Écart type sur la largeur des tournées. (∗)
S3	Envergure moyenne des tournées
S4	Écart type sur l'envergure des tournées. (∗)
S5	Profondeur moyenne des tournées.
S6	Écart type sur la profondeur des tournées. (∗)
S7	Longueur de la première et de la dernière arête de chaque tournée, divisée par la longueur totale de la tournée. (∗)
S8	Longueur moyenne de la plus grande arête de chaque tournée. (∗)
S9	Longueur de la plus grande arête de chaque tournée, divisée par la longueur de la tournée. (∗)
S10	Longueur de la plus grande arête intérieure de chaque tournée, divisée par la longueur de la tournée. (∗)
S11	Longueur moyenne de la première et de la dernière arête de chaque tournée.
S12	Demande du premier et dernier client de chaque tournée. (∗)
S13	Demande du client le plus éloigné du dépôt, pour chaque tournée. (∗)
S14	Écart type sur la demande du client le plus éloigné du dépôt. (∗)
S15	Écart type sur la longueur de chaque tournée. (∗)
S16	Distance moyenne entre les tournées
S17	Écart type sur le nombre de client de chaque tournée.
S18	Degré de chaque voisinage moyen des clients. (∗)

(∗) Caractéristiques modifiées ou nouvelles

Tableau 5.1 – Liste des caractéristiques utilisées

Test des caractéristiques

Ces caractéristiques ont été testées sur un jeu de 3000 instances générées aléatoirement, décrites dans le chapitre 4 (voir figure 4.1). Pour chaque instance, des milliers de solutions ont été créées. Celles dont le coût est moins de 1% supérieur à la meilleure solution obtenue sont considérée comme des solutions de bonne qualité. Les solutions dont

l'écart est supérieur à 2% sont considérées comme étant de mauvaise qualité. Les solutions restantes ne sont pas considérées.

Un arbre de décision a été effectué, utilisant les caractéristiques de deux tiers des solutions. Les solutions restantes sont utilisées pour tester l'arbre.

En moyenne, 95% des solutions de bonne qualité se trouvent dans une zone prometteuse. De plus, 60% des solutions dans une zone prometteuse sont de bonne qualité. Ainsi, il semble possible de focaliser les recherches sur les zones prometteuses, contenant la quasi-totalité des bonnes solutions, mais moins dense en solution de mauvaise qualité.

5.3 Des instances réalistes

5.3.1 Description des instances

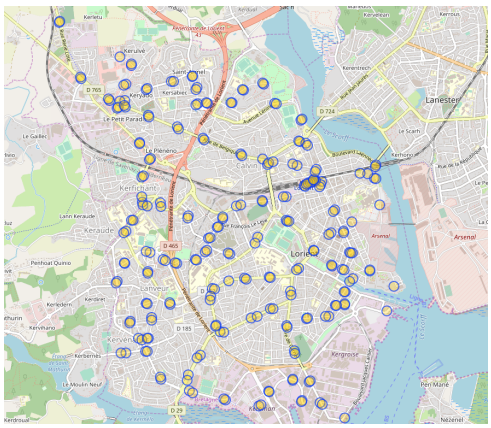
L'outil évoqué au chapitre 3 issu de la collaboration avec Gwénaël Rault et Pierre Bomel a permis de générer des instances basées sur de vraies données (clients issus de données publiques, durées de déplacement correspondant à de vrais itinéraires, etc.).

Ces instances sont issues de données récoltées sur 18 agglomérations françaises. La position du dépôt varie entre la gare et les magasins d'une grande chaîne d'ameublement, pour permettre une alternance entre dépôt centré et excentré. La position des clients est définie selon trois possibilités :

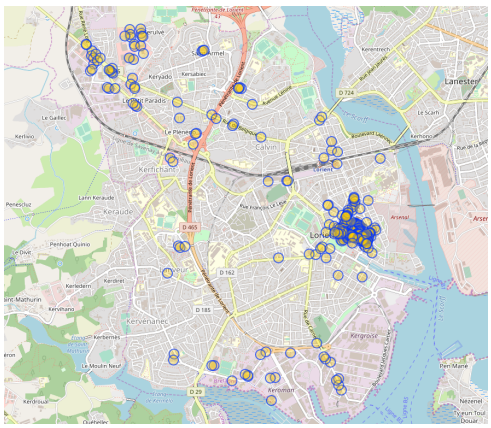
— Les arrêts de bus (voir figure 5.4(a))

— Les différents commerces (voir figures 5.4(b))

— Les différents bars et restaurants.



(a) Arrêts de bus



(b) Commerces

FIGURE 5.4 – Exemple de répartition des clients
(© les contributeurs OpenStreetMap)

Ces trois configurations permettent de tester des topologies d'instances différentes. Le nombre de clients varie de 250 à 500, par paliers de 50. La demande est définie aléatoirement par une loi normale, centrée en zéro. Si la demande est supérieure à la capacité des véhicules, des clients sont aléatoirement supprimés, jusqu'à obtenir une instance réalisable.

En combinant les différentes configurations, plus de 600 instances ont été créées.

Celles-ci ont été créées avec deux variantes, afin de tester à la fois des instances homogènes, et des instances avec trois catégories de véhicules.

5.3.2 Classification des instances

Afin de mieux comprendre si des instances similaires entraînent des solutions similaires, nous avons classé les instances. Pour cela, nous avons utilisé la classification hiérarchique, qui nous permet de comprendre plus facilement les règles définissant chaque groupe.

Caractéristiques des instances

De nouvelles caractéristiques, liées aux instances ont également été ajoutées. En particulier, deux caractéristiques, indiquant la répartition des clients et de leur demande ont été ajoutées. Les instances sont réparties géographiquement en six zones. Les trois premières zones, définies en fonction de la densité géographique sont :

1. Les zones à faible densité : elles représentent les zones sans difficulté de circulation ou de stationnement.
2. Les zones à forte densité : représentant majoritairement des territoires avec des contraintes urbaines (feux de circulations, passages piétons, etc.), la mobilité y est plus restreinte.
3. Les zones à très forte densité : représentant les hypercentres urbains, la mobilité y est très fortement réduite.

Soit G le point géographique contenant les coordonnées médianes des clients et D le dépôt, un autre partitionnement géographique a été effectué, séparant les clients selon leur distance au dépôt :

— Les clients C dont l'angle $\widehat{CGD}$ est inférieur à 45°. Cette zone correspond aux clients proches du dépôt (cône C_0 sur la figure 5.5).

— Les clients C dont l'angle $\widehat{CGD}$ est compris entre 45° et 135°. Cette zone correspond aux clients situés à une distance intermédiaire du dépôt (cônes C_1 sur la figure 5.5).

— Enfin, les clients C dont l'angle $\widehat{CGD}$ est supérieur à 135°. Cette zone correspond aux clients éloignés du dépôt (cône C_2 sur la figure 5.5).

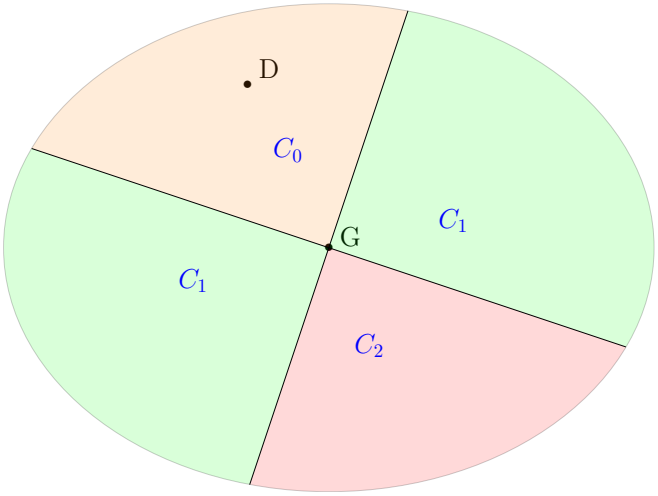


FIGURE 5.5 – Découpage géographique en cônes

Dans le chapitre 4, nous avons montré que ces caractéristiques ne sont pas utiles pour déterminer la qualité d'une solution. Cependant, elles peuvent servir à comprendre la proximité entre plusieurs instances. L'ensemble de ces caractéristiques est listé dans le tableau 5.2.

I1	Nombre de clients ;
I2	Nombre de véhicules ;
I3	Degré de capacité d'utilisation ;
I4	Distance moyenne entre les clients, pair à pair ;
I5	Écart type de la distance entre les clients ;
I6	Distance moyenne des clients vers le dépôt ;
I7	Écart type de la distance des clients vers le dépôt ;
I8	Angle moyen entre le dépôt, et les paires de clients (∗) ;
I9	Écart type sur l'angle entre le dépôt et les paires de clients ;
I10	Proportion du nombre de clients dans chacune des zones (∗) ;
I11	Proportion de la demande des clients dans chacune des zones(∗).

(∗) Nouvelles caractéristiques

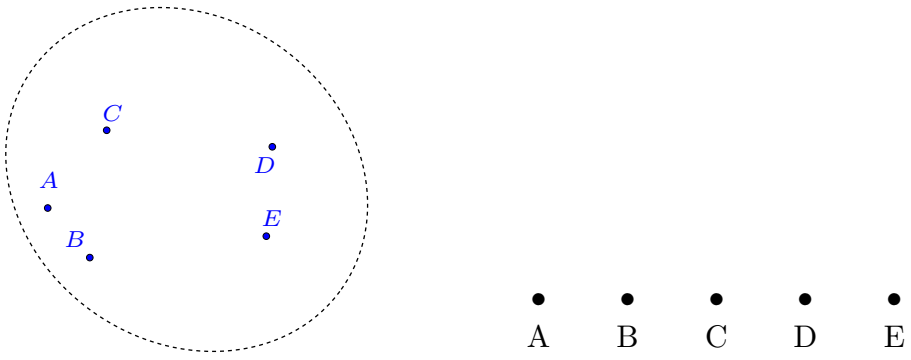
Tableau 5.2 – Caractéristiques issues des instances

Lors des expérimentations liées aux caractéristiques des instances, il s'est avéré que les caractéristiques calculant un écart type étaient très fortement corrélées à leur équivalent calculant une moyenne. Les variables d'écart type ont donc été divisées par leur équivalent en moyenne afin de réduire cette corrélation et ainsi préserver les deux catégories de variables.

La classification hiérarchique

De manière similaire à la méthode des k moyennes, la classification hiérarchique permet de regrouper les données en classes naturelles les plus homogènes possibles. Cependant, en créant un dendrogramme interprétable, la classification hiérarchique fournit plus d'informations sur la structure des groupes, et permet de choisir a posteriori le nombre de groupes.

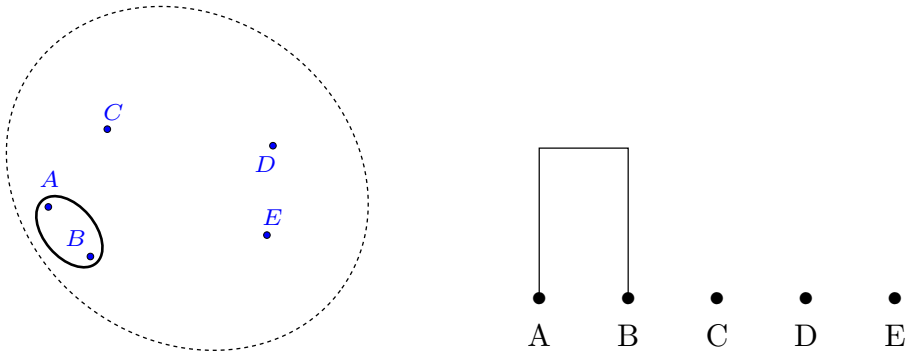
Soit un ensemble de données à séparer, comme présenté dans la figure 5.6(a), l'idée est de créer un arbre indiquant les distances entre chaque donnée.



(a) (b)

FIGURE 5.6

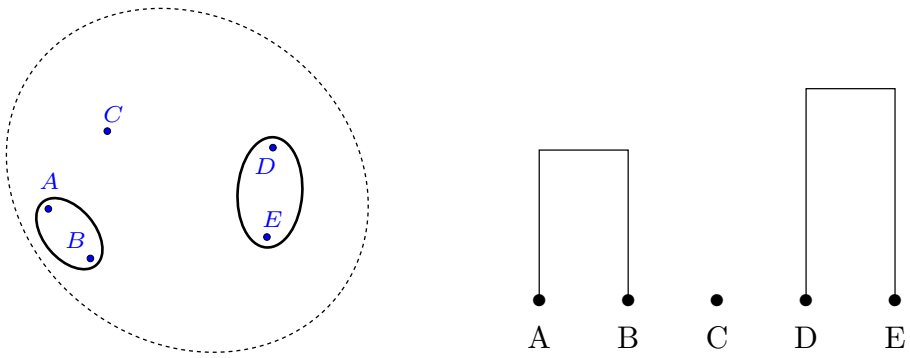
Considérons chaque solution par un groupe contenant un seul élément. L'algorithme commence par regrouper les deux données les plus proches, aussi bien dans l'espace des caractéristiques (figure 5.7(a)), que dans l'arbre (figure 5.7(b)). La hauteur du lien entre les deux éléments dans l'arbre correspond à la distance les séparant dans l'espace des caractéristiques.



(a) (b)

FIGURE 5.7

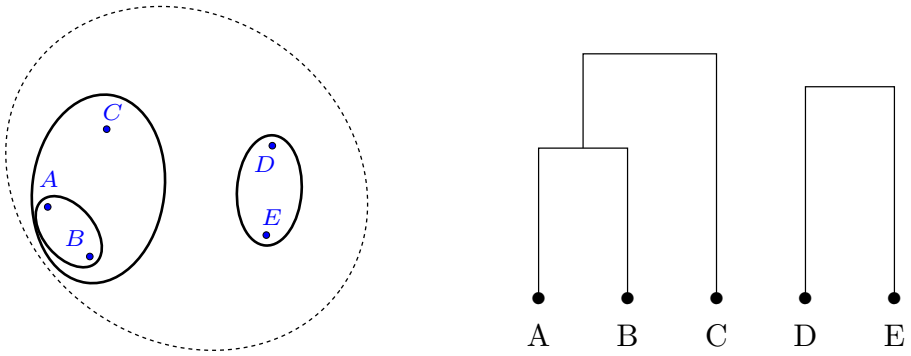
L'algorithme met ensuite à jour les distances. En effet, ici seules les distances entre chaque groupe sont prises en compte. Soit $d_{i,j}$ la distance entre deux données i et j. Soit G_1 et G_2 deux groupes de données. La distance entre deux groupes est ici définie par la plus petite distance entre les éléments de chaque groupe : $\min_{i \in G_1} \min_{j \in G_2} d_{i,j}$.



(a) (b)

FIGURE 5.8

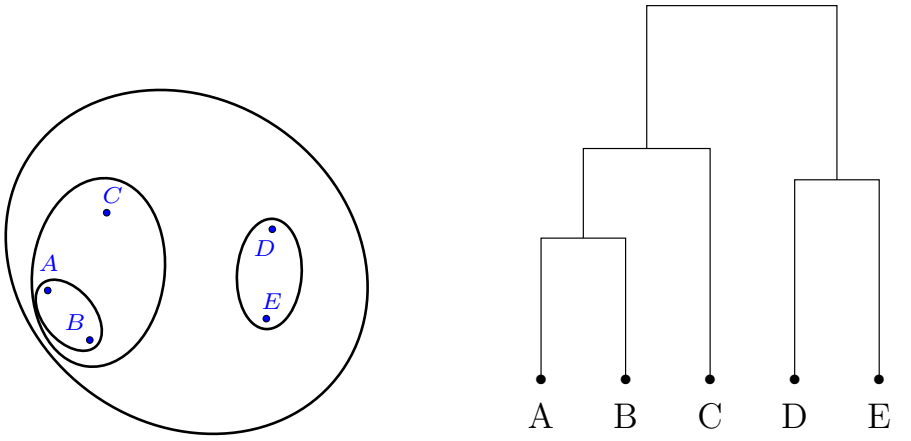
Dans notre exemple, la nouvelle plus petite distance concernent les données D et E (voir figure 5.8(a)). L'arbre est à nouveau mis à jour (figure 5.8(b)).



(a) (b)

FIGURE 5.9

Ensuite, la plus petite distance est celle entre la donnée C et le groupe $\{A, B\}$ (figure 5.9(a)). Ce regroupement est indiqué dans l'arbre (figure 5.9(b)).



(a) (b)

FIGURE 5.10

Enfin, les deux groupes restants sont combinés pour ne former qu'un seul groupe (figure 5.10).

L'arbre étant terminé, il est possible de faire un partitionnement des données. Il suffit de séparer les données par ordre décroissant de distance, comme indiqué dans l'arbre. Ainsi, un partitionnement en deux groupes donne les ensembles $\{A, B, C\}$ et $\{D, E\}$, alors qu'un partitionnement en trois groupes donne les trois ensembles suivants : $\{A, B\}$, $\{C\}$ et $\{D, E\}$.

Regroupement des instances

Instances à flotte homogène La classification hiérarchique pour les instances à une catégorie de véhicules a donné l'arbre de la figure 5.11. Le but de cette identification est de séparer les instances en sous-groupes identifiables et de taille similaire. Le nombre de groupes doit donc être faible pour pouvoir aisément les étudier séparément. Après observation de l'arbre, le meilleur moyen d'obtenir un nombre restreint de groupe, de taille similaire, est un découpage en trois groupes, comme indiqué dans la figure 5.11.

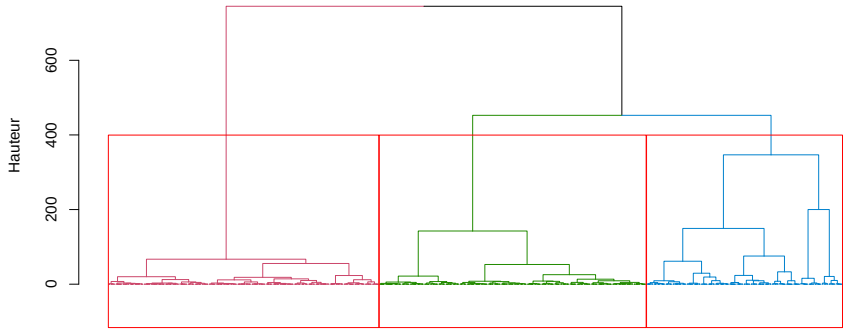


FIGURE 5.11 – Classification hiérarchique des instances

Pour mieux interpréter les différentes classes d'instance, un arbre de décision a été effectué (figure 5.12). Celui-ci permet de générer des règles simples et faciles à interpréter pour mieux comprendre ce qui crée les identités de chaque groupe créé dans l'arbre de la figure 5.11. La quasi-totalité des instances a été correctement classée, les règles que l'arbre décrit sont donc fiables.

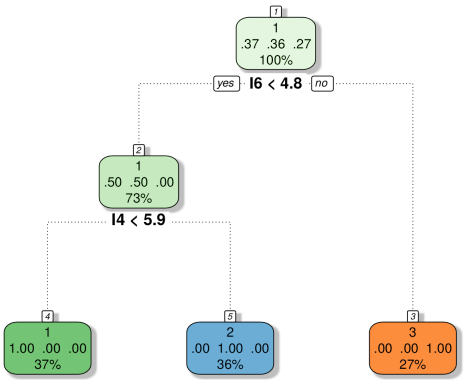


FIGURE 5.12 – Règles de séparation des instances

Le premier groupe est défini par les règles suivantes :

— La distance moyenne entre les clients et le dépôt est < 4.8

— La distance entre les clients est < 5.9.

Les clients sont donc proches entre eux, et proches du dépôt. Ce sont donc des instances où les clients sont concentrés autour du dépôt (voir figure 5.13).

Les figures 5.13 à 5.16 sont des représentations graphiques de certaines des instances réelles traitées. Pour chacune de ces figures, le point en rouge correspond au dépôt et le point en vert correspond aux coordonnées médianes des clients.

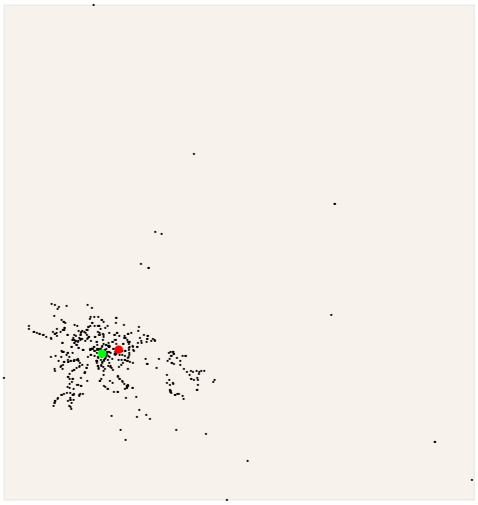


FIGURE 5.13 – Exemple d'instance du groupe 1

Dans le deuxième groupe d'instances, les clients sont toujours proches du dépôt, mais leur distance deux à deux est plus grande. Il s'agit donc d'instances dont le dépôt est central, mais où les clients sont répartis sur une plus large zone.

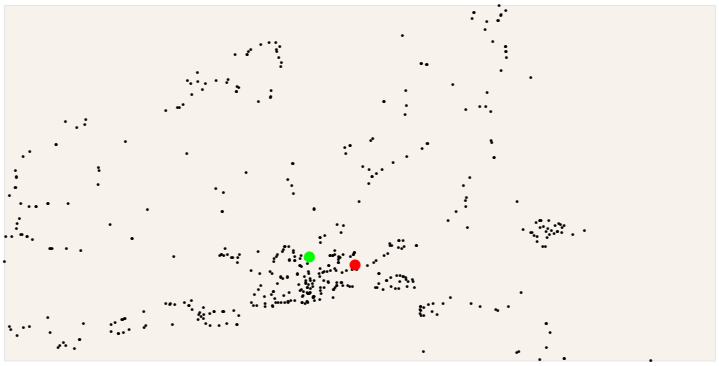


FIGURE 5.14 – Exemple d'instance du groupe 2

Dans le troisième groupe, la distance avec le dépôt est élevée. Il peut s'agir d'instances avec un nombre important de clients à la périphérie (figure 5.15), ou d'instances dont le dépôt est excentré (figure 5.16).

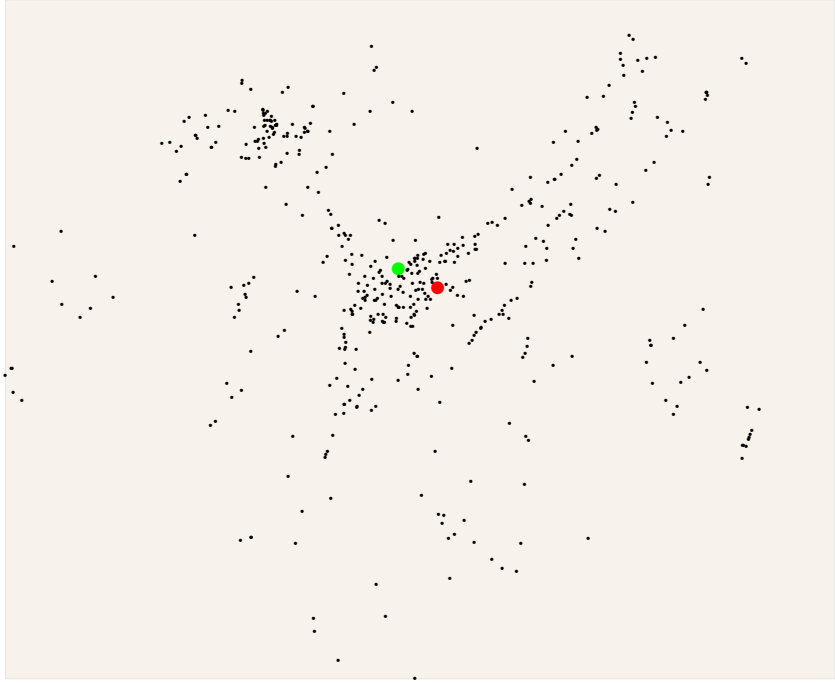


FIGURE 5.15 – Exemple d'instance du groupe 3 (dépôt centré)

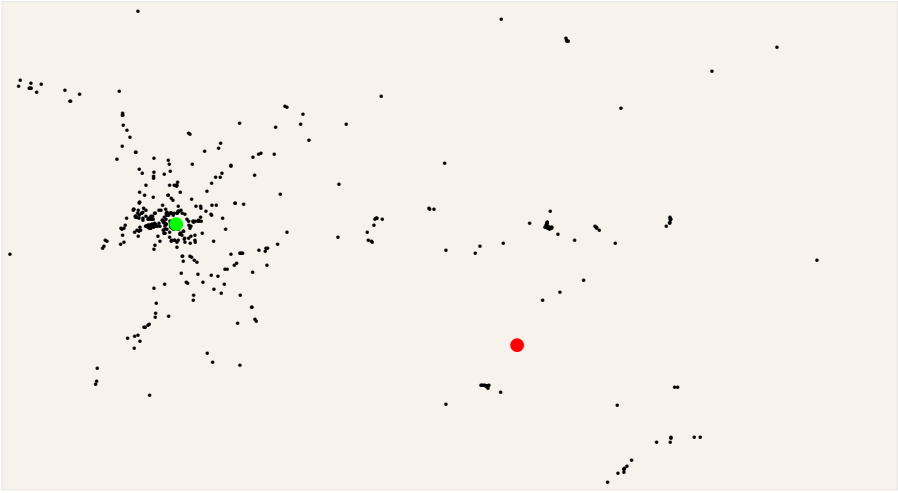


FIGURE 5.16 – Exemple d'instance du groupe 3 (dépôt excentré)

Instances à flotte hétérogène Dans le cas des instances avec plusieurs catégories de véhicules, la classification hiérarchique a permis d'identifier quatre groupes d'instances. Les règles séparant les groupes, sont similaires aux règles de différenciation pour les instances à flotte homogène (voir figure 5.17). La différence importante concerne les instances dont la distance au dépôt est élevée.

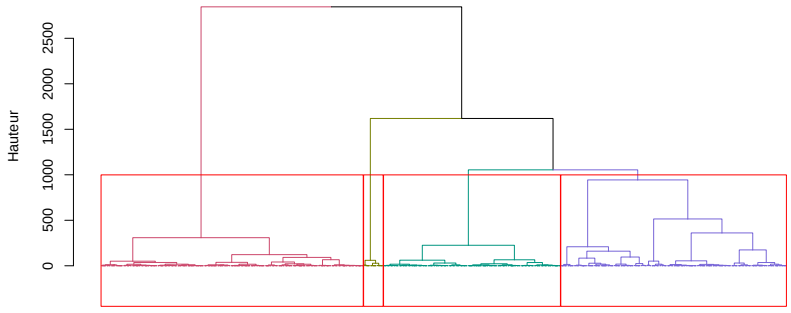


FIGURE 5.17 – Classification hiérarchique des instances à flotte hétérogène

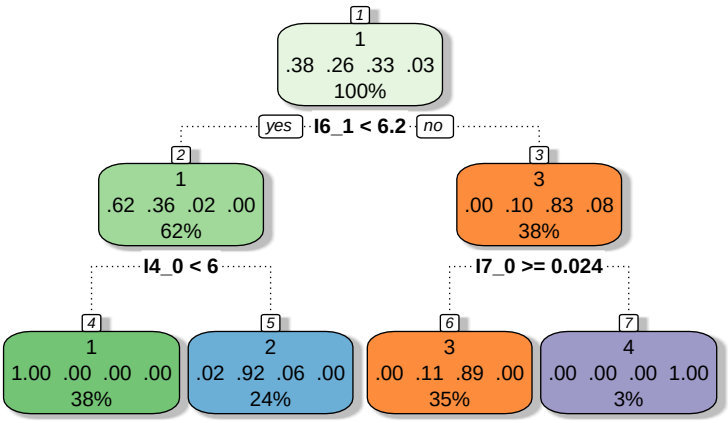


FIGURE 5.18 – Règles de séparation des instances à flotte hétérogène

Dans ce groupe d'instances, il y a un sous-groupe qui se différencie, en fonction de l'écart type de la distance au dépôt. Ces instances sont toutes liées à la même ville et possèdent le même dépôt, qui est extrêmement excentré (voir figure 5.19). Les variables d'écart type étant normalisées avec les variables moyennes associées, cela s'est traduit par une faible valeur de la variable d'écart type. La classification hiérarchique a donc permis d'identifier un petit groupe d'instances avec une topologie particulière.

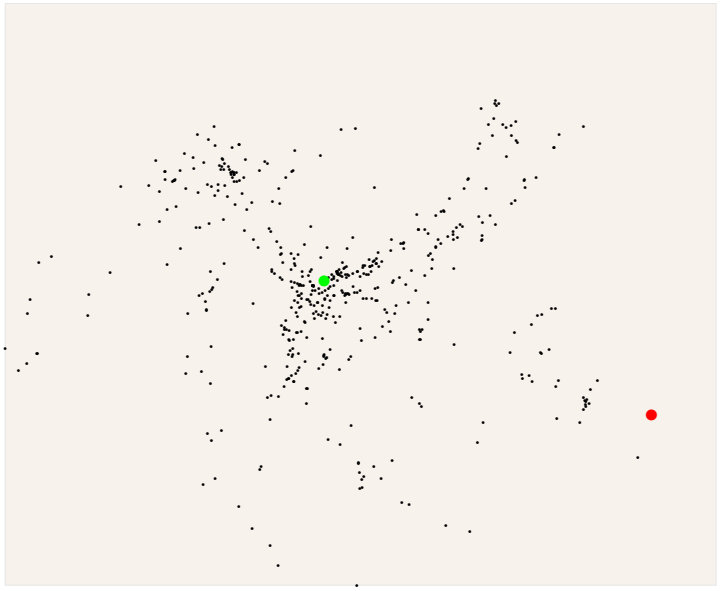


FIGURE 5.19 – Exemple d'instances avec un dépôt très excentré

5.4 Expérimentations

5.4.1 Protocole expérimental

Plusieurs versions de l'algorithme FG-MNS sont testées, en plus d'une version témoin sans apprentissage automatique et d'une version servant uniquement pour l'apprentissage hors ligne. En dehors de cette dernière version, le critère d'arrêt est toujours d'une minute. Le FG-MNS est implémenté en C++, à l'exception du calcul des arbres de décision, qui est effectué en R.

Sept modes ont été exécutés :

Mode 0 - Solveur témoin Il s'agit du solveur RADOS décrit dans le chapitre 3, qui n'utilise pas d'apprentissage automatique. Le solveur s'arrête après une minute de calcul.

Mode 1 - Base d'apprentissage Ce mode correspond au solveur RADOS du chapitre 3, avec trois voisinages avec chaînes d'éjection supplémentaires : la réallocation inter-routes, et les path-moves de tailles 2 et 3. Les caractéristiques de chacune des solutions observées sont calculées et stockées. Le solveur s'arrête au bout de 10 minutes et retourne les caractéristiques de 10 000 solutions parmi celles stockées.

Les données ainsi récoltées vont servir à la création des arbres de décision pour les différentes méthodes hors ligne testées. Afin de séparer les données utilisées pour générer les règles de celles servant à vérifier leur fiabilité, les instances et les données de leurs solutions associées sont réparties aléatoirement en plusieurs groupes. Il est fréquent en apprentissage automatique d'attribuer les deux tiers des données pour l'entraînement et le tiers restant pour tester les règles générées.

En reprenant ce principe, nous avons décidé de créer trois groupes aléatoires d'instances G_1, G_2, G_3. Ainsi, lorsqu'une instance issue du groupe G_1 doit être résolue avec une méthode guidée par un apprentissage hors ligne sans classification hiérarchique, les données utilisées pour créer l'arbre proviendront des solutions des instances des groupes G_2 et G_3. De manière similaire, les instances issues du groupe G_2 (respectivement G_3) seront guidée par un arbre entraîné sur les instances de G_1 et G_3 (respectivement G_1 et G_2).

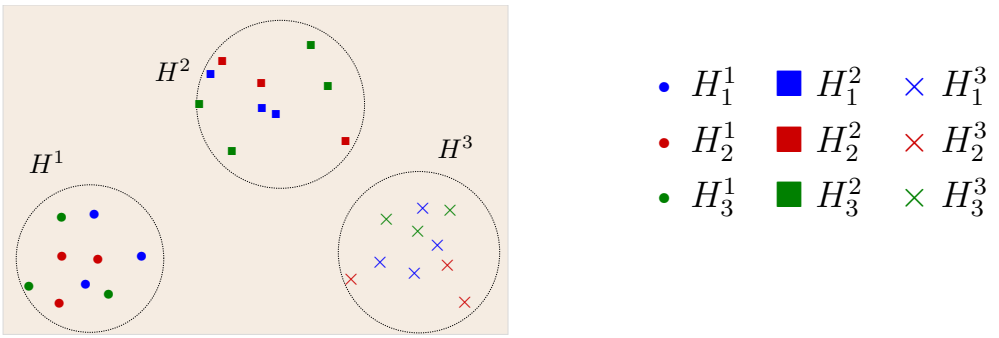


FIGURE 5.20 – Exemple de sous-groupes d'instances

Soit H^i le i^{eme} groupe d'instances issu de la classification hiérarchique, celui-ci est divisé en trois sous-groupes H_1^i, H_2^i et H_3^i, comme indiqué dans la figure 5.20. Ainsi, de manière équivalente à l'apprentissage hors ligne sans classification hiérarchique, un solveur guidé par un apprentissage hors ligne avec utilisation de classification hiérarchique dont l'instance courante est issue de H_1^i sera guidé par un arbre utilisant les données issues de $H_2^i \cup H_3^i$.

Mode 2 - Apprentissage en ligne Le solveur s'exécute comme le solveur RADOS du chapitre 3 pendant 20s. Pendant cette phase d'apprentissage, les caractéristiques de toutes les solutions sont stockées. Un arbre de décision est ensuite exécuté. Une fois les zones prometteuses définies, la version guidée du solveur s'exécute, jusqu'à atteindre une durée totale de 60s. La construction de l'arbre de décision durant en moyenne 10s, la version guidée dure donc environ 30s.

Mode 3 - Apprentissage hors ligne Soit G_i le groupe dont est issue l'instance à résoudre, le solveur FG-MNS s'exécute en étant guidé par l'arbre de décision issu de $G \backslash G_i$, puis s'arrête après une minute d'exécution.

Mode 4 - Apprentissages hors ligne et en ligne Soit G_i le groupe dont est issue l'instance à résoudre, le solveur FG-MNS s'exécute en étant guidé par l'arbre de décision issue de $G \backslash G_i$, puis s'arrête après 20s d'exécution. Pendant cette première phase, les caractéristiques de toutes les solutions sont stockées. Un second arbre de décision, basée uniquement sur les solutions de l'instance en cours de d'exécution est ensuite généré. Une fois les zones prometteuses définies par l'arbre, la version guidée du solveur s'exécute, jusqu'à atteindre une durée totale de 60s.

Mode 5 - Classification et apprentissage hors ligne Soit H^i_j le sous-groupe dont est issue l'instance à résoudre, le solveur FG-MNS s'exécute en étant guidé par l'arbre de décision issue de $H^i \backslash H^i_j$, puis s'arrête après une minute d'exécution.

Mode 6 - Classification et apprentissages hors ligne et en ligne Soit H^i_j le sous-groupe dont est issue l'instance à résoudre, le solveur FG-MNS s'exécute en étant guidé par l'arbre de décision issue de $H^i \backslash H^i_j$, puis s'arrête après une minute d'exécution. Pendant cette première phase, les caractéristiques de toutes les solutions sont stockées. Une fois les zones prometteuses définies, la version guidée du solveur s'exécute, jusqu'à atteindre une durée totale de 60s.

5.4.2 Résultats

Les résultats des expérimentations sont disponibles dans le tableau 5.3. Pour chaque instance, la meilleure solution obtenue en appliquant les 7 différents modes est celle servant de référence pour les calculs des écarts moyens et médians. Ces solutions ont été presque intégralement obtenues lors de l'application du mode 1, servant de base aux méthodes d'apprentissage hors ligne. Les arbres de décision qui en résultent servent donc à guider la recherche vers des solutions de meilleure qualité que le solveur témoins (mode 0).

Cependant, aucune variante du FG-MNS n'a réussi à être significativement plus performante que RADOS. Concernant les instances homogènes, le mode le plus efficace utilise uniquement un apprentissage hors ligne, basé sur des instances classées par similarités.

La classification hiérarchique semble donc une bonne piste pour améliorer les règles de décision.

Mode	Temps CPU	Instances homogènes		Instances hétérogènes	
		Écart moyen (%)	Écart médian (%)	Écart moyen (%)	Écart médian (%)
Mode 0	60	0.71	0.65	2.13	2.04
Mode 1	600	0.05	0.00	0.01	0.00
Mode 2	60	1.03	0.94	2.54	2.51
Mode 3	60	1.11	0.91	2.31	2.30
Mode 4	60	1.11	0.99	2.52	2.52
Mode 5	60	0.92	0.82	2.62	2.57
Mode 6	60	1.10	1.02	2.78	2.70

Tableau 5.3 – Résultats des expérimentations du FG-MNS

Concernant les instances à flotte hétérogène, le solveur FG-MNS est plus performant lorsqu'il utilise l'apprentissage hors ligne, guidé par des arbres dont les données d'entrée sont issues d'instances non classées.

5.5 Analyse de résultats

Le FG-MNS n'est pas aussi performant que souhaité. Différentes expérimentations supplémentaires ont été effectuées pour comprendre l'origine de ces résultats.

5.5.1 Étude des arbres de décision

Un solveur basé sur des arbres de décision ne peut fonctionner que si ceux-ci sont performants. Différentes études ont donc été effectuées pour voir s'il est possible d'obtenir de meilleurs arbres.

Paramètres de l'arbre de décision

Les arbres de décision utilisés pour les apprentissages en ligne et hors ligne ont été exécutés avec des paramètres bien précis :

Taille des données Les arbres de décision pour l'apprentissage hors ligne utilisent les données de 10.000 solutions par instance de l'ensemble d'entraînement. Les arbres de décision pour l'apprentissage en ligne utilisent les données des solutions parcourues

pendant la phase d'apprentissage du FG-MNS, avec une limitation de 10 000 solutions, pour réduire le temps de fabrication de l'arbre à environ 10 secondes.

Séparation des solutions Pour apprendre à séparer les bonnes solutions des mauvaises, il faut une définition précise les séparant. Afin d'avoir une stabilité dans la qualité des solutions classées, les médianes ont été utilisées. Les solutions sont triées par coûts croissants, les 1% meilleures sont classées comme bonnes, les 85% moins bonnes sont classées comme mauvaises et les 14% restantes sont éliminées, afin de garantir une différence nette entre les solutions de chaque catégorie.

Équilibre des classes La séparation des solutions ainsi obtenue implique qu'il y a 85 fois plus de mauvaises solutions que de bonnes. Ainsi, un arbre prédisant que toutes les solutions sont de mauvaise qualité est incapable de repérer les bonnes solutions mais a un taux de réussite de 98.8%. Pour éviter un arbre déséquilibré, les bonnes solutions ont une pondération plus importante, calculée pour que la somme des poids de chacun des ensembles (bonnes solutions, mauvaises solutions) des données d'entraînement soit égale.

Profondeur des arbres Plus un arbre est profond, plus il crée de règles. Ces règles permettent d'affiner les frontières entre les deux catégories de solutions, dans les données d'entraînement. En contrepartie, il y a toujours un risque de créer des règles trop spécifiques qui ne fonctionnent que sur les données d'entraînement. Cette recherche d'équilibre s'appelle le *dilemme biais-variance*. Les arbres de décision sont des méthodes classiques de l'apprentissage automatique et les fonctions développées créent des arbres équilibrés.

Cependant, dans le cadre du FG-MNS, un arbre trop profond signifie que les zones prometteuses associées sont définies par un nombre important de caractéristiques. Ainsi, lors du guidage d'une solution vers une de ces zones prometteuses, la recherche locale devra, pour chacune des solutions du voisinage de recherche, calculer un nombre important de caractéristiques. Pour éviter des temps de calcul trop importants, la profondeur des arbres a été arbitrairement limitée à 5.

Recherche des paramètres

Afin d'étudier la qualité des arbres, plusieurs paramètres ont été modifiés dans le but de voir s'ils influent sur la qualité de l'arbre. Nous nous sommes ici concentrés sur la

performance des arbres pour les données issues des solutions des instances homogènes du jeu d'instances réalistes.

Données d'entraînement Le solveur RADOS a de nouveau été exécuté sur chaque instance, afin de récolter des solutions. Celui-ci s'arrête dès qu'il a récolté 20 000 solutions différentes, ce qui correspond à un temps d'exécution compris entre 5 et 10 secondes, selon les instances. Ces données sont ensuite filtrées : une seconde élimination de doublons (dus à des erreurs d'arrondis) est effectuée. Le coût des solutions est converti en écart par rapport au coût de la meilleure solution de ce jeu de données, permettant ainsi de normaliser la valeur de chaque solution.

Critère de séparation Les performances du solveur étant globalement uniformes sur le jeu d'instances réelles, le critère de séparation a donc été fixé sur des valeurs de l'écart à la meilleure solution du jeu de données, et non sur une médiane. Les solutions dont l'écart est inférieur à 1% sont classées en solution de bonne qualité, celles avec un écart supérieur à 1.5% sont considérées de mauvaise qualité. Les autres solutions sont éliminées des jeux de données.

Taille des données Le nombre de solutions étudiées par instance fait partie des paramètres étudiés. Ce paramètre varie de 300 à 2000. Les solutions sont tirées aléatoirement parmi les solutions filtrées précédemment. Afin de limiter les déséquilibres entre les classes de solutions, chaque jeu de données contient, lorsque cela est possible, un tiers de solution de bonne qualité et deux tiers de solutions de mauvaise qualité.

Équilibrage des classes Les jeux de données sont donc toujours en partie déséquilibrés. Afin de résoudre ce problème, deux variantes d'équilibrage ont été testées :

— La pondération des solutions : les solutions de bonne qualité seront considérées avec deux fois plus d'importance que les solutions de mauvaise qualité. Cela revient à dupliquer ces solutions.

— La méthode SMOTE : cette méthode consiste à relier chaque solution de bonne qualité à ses k solutions voisines les plus proches, puis à générer des points factices sur ces arêtes jusqu'à obtenir autant de solutions (réelles ou factices) de chaque classe (voir figure 5.21). Par défaut, k est fixé à 5.

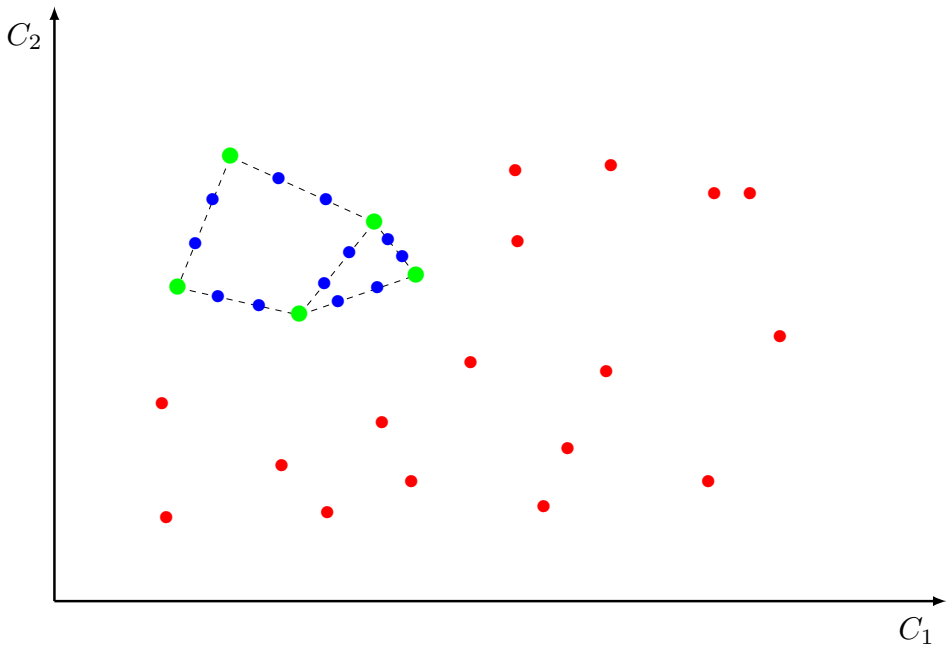


FIGURE 5.21 – Génération de données par l'algorithme SMOTE

Dans la figure 5.21, les solutions de bonne qualité (en vert) sont moins représentées que les solutions de mauvaise qualité (en rouge). Des solutions factices (en bleu) sont créées jusqu'à obtenir autant de solutions de bonne qualité (factices ou réelles) que de solution de mauvaise qualité.

Profondeur de l'arbre Nous avions précisé précédemment que la profondeur maximale de l'arbre a été fixée à 5 pour limiter le temps de la recherche guidée. Ici, nous avons testé les performances d'arbre à différentes profondeurs, que nous avons fait varier de 1 à 10. Nous avons également testé des arbres sans limite de profondeur, laissant les fonctions pré-implémentées en R arrêter l'arbre lorsque les performances cessent de croître.

Nombre de classes d'instances Le tableau 5.3 semble indiquer que la classification hiérarchique est prometteuse pour améliorer le guidage. Nous avons donc étudié l'équilibre quantité/qualité des données, en faisant varier le nombre de groupes d'instances. Ce nombre varie de 1 (pas de classification hiérarchique) à 38 (peu de données pour chaque arbre, mais issues d'instances très proches). Chaque classe d'instance H^i est divisé en trois sous-groupes H^i_1, H^i_2 et H^i_3 tels que $H^i = H^i_1 \cup H^i_2 \cup H^i_3$. La qualité des solutions appartenant aux instances du sous-groupe H^i_j, $j \in \{1, 2, 3\}$ est prédite à partir d'arbres de décision provenant de $H^i \backslash H^i_j$.

Critères de performances Nous avons vu plus tôt qu'évaluer la performance d'un arbre de décision en fonction de sa performance moyenne (i.e. la proportion de bonnes prédictions) n'est pas un critère suffisant. Dans la littérature, trois critères sous souvent utilisés lorsque les classes sont déséquilibrées.

Notons $t = \begin{bmatrix} t_{00} & t_{01} \\ t_{10} & t_{11} \end{bmatrix}$ la matrice, appelée *matrice de confusion*, indiquant la répartition des prédictions en fonction de la qualité des solutions. Les lignes correspondent aux *observations* (la qualité effective des solutions), tandis que les colonnes correspondent aux prédictions. Ainsi, t_{00} représente le nombre de vrais négatifs, t_{01} le nombre de faux positifs, t_{10} le nombre de faux négatifs et t_{11} le nombre de vrais positifs.

— *Taux de réussite* : ce critère indique la proportion prédictions de correctes parmi l'ensemble des prédictions : $\frac{t_{00}+t_{11}}{t_{00}+t_{01}+t_{10}+t_{11}}$.

— *Précision* : Ce critère correspond à la proportion de vrais positifs parmi les prédictions positives : $\frac{t_{11}}{t_{01}+t_{11}}$. Dans le cadre du FG-MNS, ce nombre correspond à la proportion de bonnes solutions dans les zones prometteuses.

— *Rappel* : Ce critère est égal au rapport entre le nombre de vrais positifs et le nombre de données positives : $\frac{t_{11}}{t_{10}+t_{11}}$.
Il correspond ici à la proportion de solutions de bonne qualité qui seront observées en ne se focalisant que sur les zones prometteuses.

— *F1* : Ce critère correspond à la moyenne harmonique entre les deux critères précédents. Il permet d'avoir un équilibre entre la densité en bonnes solutions des zones prometteuses, et le nombre de solutions intéressantes laissées de côté.

Chaque critère varie donc de 0 (aucune prédiction correcte) à 1 (toutes les prédictions sont correctes) et doit donc être maximisé.

Protocole expérimental La totalité des combinaisons possibles a été testée, à l'exception de certaines combinaisons avec les jeux de données comportant plus de 1000 solutions par instance, afin de limiter le temps d'obtention des résultats. Ainsi, les arbres de décision sur les jeux de données de grande taille n'ont pas été testés sur des classifications hiérarchiques à plus de trois groupes d'instances. Les performances des arbres ont été évaluées selon les données en entrée, appelées *ensemble d'apprentissage*, avant d'être testées sur d'autres données, appelées *ensemble de test*, comme précisé au paragraphe précédent.

Performances sur les données d'entraînement Si on ne considère que l'ensemble d'entraînement, les différents critères obtiennent des performances proches de 0.90 avec les bons jeux de paramètres (voir tableau 5.4). L'arbre de décision semble donc prometteur, au moins pour l'apprentissage en ligne.

	Taux de réussite	Rappel	Précision	F1
Minimum	0,43	0,44	0,35	0,43
Maximum	0,92	0,94	0,90	0,92

Tableau 5.4 – Valeurs limites des critères sur les différents jeux de paramètres

Jeux de paramètres optimaux Après étude des résultats, le jeu de paramètre optimal n'est pas le même selon le critère de performance choisi. Nous les étudierons donc séparément. Les tableaux 5.5 et 5.6 présentent les meilleurs jeux de paramètres selon le taux de réussite et le critère de rappel, ainsi que les valeurs associées pour les autres critères.

Taux de réussite Sur les données d'entraînement, Les meilleurs jeux de paramètres pour ce critère, disponibles dans le tableau 5.5 donnent un taux de réussite maximal de 62.3%, ce qui n'est pas exceptionnel. Pour maximiser le taux de réussite des arbres de décision, il semble nécessaire d'augmenter la taille des données, le nombre de groupes d'instances, et de ne pas limiter la taille de l'arbre. L'utilisation de la méthode SMOTE semble n'avoir aucun impact sur cet indicateur de performance.

Solutions / instances	Nombre de groupes	Utilisation du SMOTE	Profondeur maximum	Taux de réussite	Rappel	Précision	F1
1000	38	FALSE	∞	0,623	0,397	0,429	0,413
1000	38	TRUE	∞	0,621	0,398	0,426	0,412
750	26	TRUE	∞	0,621	0,394	0,425	0,410
1000	26	TRUE	∞	0,620	0,407	0,426	0,417
750	38	TRUE	∞	0,620	0,387	0,424	0,405
1000	12	TRUE	∞	0,620	0,427	0,429	0,428
750	38	FALSE	∞	0,620	0,400	0,425	0,413
1000	38	FALSE	10	0,619	0,407	0,425	0,416
1000	26	TRUE	10	0,619	0,410	0,425	0,417
1000	38	TRUE	10	0,618	0,406	0,424	0,415
...	...	...	...	...	...	...	...

Tableau 5.5 – Meilleurs paramètres pour maximiser le taux de réussite

Critère de rappel Conformément au tableau 5.6, pour maximiser le critère de rappel, il ne faut pas séparer les instances, et limiter le nombre de règles. La taille des données semble à nouveau permettre d'améliorer les performances, tandis que l'algorithme SMOTE semble toujours aussi peu utile. Enfin, les critères de taux de réussite et de rappel semblent fortement anti-corrélés.

Ces observations laissent entendre que d'une instance à l'autre, les valeurs des caractéristiques des bonnes solutions varient trop pour pouvoir créer des règles fiables.

Solutions / instances	Nombre de groupes	Utilisation du SMOTE	Profondeur maximum	Taux de réussite	Rappel	Précision	F1
2000	01	FALSE	1	0,431	0,859	0,349	0,604
1750	01	FALSE	1	0,433	0,859	0,351	0,605
2000	01	TRUE	1	0,433	0,856	0,349	0,603
1500	01	FALSE	1	0,443	0,840	0,355	0,598
1500	01	TRUE	1	0,443	0,840	0,355	0,598
1750	01	TRUE	1	0,442	0,840	0,353	0,597
1250	01	FALSE	1	0,444	0,839	0,356	0,598
300	01	FALSE	1	0,446	0,839	0,359	0,599
300	01	TRUE	1	0,446	0,839	0,359	0,599
1000	01	FALSE	1	0,444	0,838	0,357	0,598
500	01	FALSE	1	0,446	0,837	0,358	0,597
500	01	TRUE	1	0,446	0,837	0,358	0,597
...	...	...	...	...	...	...	...

Tableau 5.6 – Meilleurs paramètres pour maximiser le critère de rappel

Critère de précision et F1 Le critère de précision est très fortement corrélé au taux de réussite et peut atteindre 42.9%. Les observations sont donc les mêmes que pour le taux de réussite.

Supposons la règle qui considère l'espace des caractéristiques en entier comme une zone prometteuse, on obtient un score de 33.0% (car il y a 33% de bonnes solutions dans les jeux de données), le gain est donc effectif, mais peu significatif.

De façon similaire, le critère F1 est très fortement corrélé au critère de rappel. Il peut atteindre 60.5% avec la bonne configuration.

Graphe de corrélations Afin de vérifier les hypothèses liant les paramètres et les critères de performances, un graphe de corrélation a été créé (voir figure 5.22). On y remarque que :

— les critères de rappel et de précision sont fortement anti-corrélés,

— l'algorithme SMOTE n'est corrélé avec aucun critère,

— le critère de précision est corrélé avec le nombre de groupes et la profondeur maximum de l'arbre,

— le nombre de solutions semble finalement peu corrélé avec les critères.

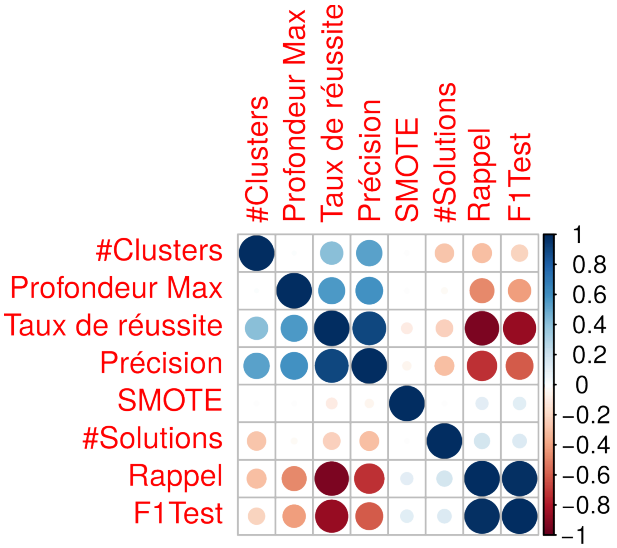


FIGURE 5.22 – Graphe de corrélation entre les caractéristiques et critères

À l'exception du paramètre sur le nombre de solutions, les hypothèses précédentes sont vérifiées par le graphe de corrélation.

Déductions Les arbres créés sont donc peu performants, cependant, quelques points semblent concorder :

1. Le critère de rappel décroît avec le nombre de nœuds de l'arbre de décision. Les bonnes solutions sont donc suffisamment dispersées pour qu'à chaque coupe dans l'espace des caractéristiques, une partie d'entre elles se retrouve dans une zone contenant majoritairement des solutions de mauvaise qualité.
2. Inversement, le critère de précision s'améliore avec le nombre de coupes. Ainsi, malgré la perte de bonnes solutions à chaque coupe, les zones prometteuses gagnent en densité. Certaines zones très précises semblent donc être riches en bonnes solutions.
3. Enfin, nous avons vu que créer des solutions factices entre deux solutions voisines de bonne qualité (via l'algorithme SMOTE) au lieu d'augmenter le poids des bonnes solutions ne changeait pas les performances des arbres de décision. Cette observation laisse supposer que les bonnes solutions sont disposées en îlots.

En regroupant les trois informations, nous pouvons supposer que les solutions sont regroupées en îlots, de densités variables : les groupes de solutions les plus faibles étant éliminées par les coupes, tandis que les contours des groupes les plus denses sont affinés au fur et à mesure des coupes. Ces îlots étant constitués de solutions extrêmement similaires, il s'agit probablement des solutions constituant les différents minima locaux parcourus.

Piste d'amélioration La figure 5.22 nous indique que le nombre de groupes d'instances est corrélé avec la performance globale des arbres. Des instances similaires donnent donc des solutions similaires. Cependant, même avec 38 groupes d'instances, les arbres restent peu performants. Il semble donc nécessaire d'améliorer la classification des instances. Un problème de classification peut venir des données, ou de la méthode utilisée. La classification hiérarchique est une méthode réputée pour son efficacité et sa robustesse. Ainsi, une piste prometteuse pour améliorer l'efficacité des arbres de décision, est de trouver un meilleur ensemble de caractéristiques liées aux instances. Ces meilleures caractéristiques permettront de mieux comprendre les instances, et donc de mieux les classer. Les arbres de décision seront ainsi créés sur des instances similaires, augmentant leur efficacité.

5.5.2 Expérimentations avec un arbre oracle

Nous avons vu précédemment qu'il est difficile d'obtenir des arbres de décision de qualité, pour de l'apprentissage hors ligne. Ici, nous allons voir si améliorer les arbres actuels

(par exemple en améliorant la classification des instances), en gardant les caractéristiques des solutions telles que définies en 5.1 permet d'améliorer le FG-MNS.

Pour cela, nous avons créé des arbres oracles propres à chaque instance. Ces arbres sont créés à partir de données récoltées sur l'instance à tester lors de l'exécution de RADOS complété par des voisinages à chaînes d'éjections pendant 10 minutes (voir mode 1, page 131). Les arbres ainsi obtenus classifient donc également des solutions de coût jusqu'à 0.7% inférieur en moyenne, que le solveur RADOS sans chaîne d'éjection (voir mode 0 131).

Le solveur FG-MNS utilisant uniquement de l'apprentissage hors-ligne a ensuite été exécuté sur chacune des instances. L'écart de coût moyen (respectivement médian) obtenu sur la meilleure solution de chaque instance est de 1.25% (respectivement 1.06%), ce qui est nettement moins performant que RADOS (écarts-moyens et médians de 0.71% et 0.65%). De nombreuses variantes ont été testées (changement du critère de classifications des solutions, variation de la zone prometteuse cible, etc.), sans réussir à améliorer drastiquement les performances.

Il semblerait donc qu'en l'état actuel, le FG-MNS ne soit pas apte à fournir de meilleures performances qu'une méthode sans apprentissage tel qu'utilisé par le solveur RADOS.

Pistes d'améliorations

Plusieurs hypothèses sont à prendre en compte concernant les performances du FG-MNS :

Le choix des caractéristiques Il est possible que certaines caractéristiques indispensables à la définition d'une solution n'aient pas encore été découvertes. Ajouter de nouvelles caractéristiques permet d'ajouter des informations sur l'ensemble des solutions à étudier, et donc d'affiner les règles qui en sont tirées.

Diversité des solutions Le FG-MNS est basé sur des méthodes de recherche locale. Les solutions à améliorer issues d'un GRASP sont censées être suffisamment distinctes pour permettre d'avoir un aperçu de la topologie de l'instance à résoudre. Tester des arbres de décisions basées sur des solutions créées par des méthodes forçant la diversification des solutions, comme le UHGS (Vidal *et al.*, 2014) permettrait probablement d'améliorer la compréhension du lien entre les caractéristiques d'une solution et sa qualité.

5.6 Conclusion

Une méthode hybride, associant recherche opérationnelle et méthodes d'apprentissage automatique, nommée FG-MNS a été proposée et testée sur un nouveau jeu d'instances issues de données réelles. Malgré nos tests étendus, le FG-MNS, à temps équivalent, ne rivalise pas avec un solveur aussi performant que RADOS.

Les différentes expérimentations menées ont permis de mieux comprendre ce qui constitue une bonne solution, approfondissant ainsi nos connaissances sur la question de recherche 2 *comment caractériser une solution d'un problème de tournée ?*. En ne restant focalisé que sur l'amélioration de la fonction objectif, la question de recherche 3 *comment guider la recherche d'une bonne solution grâce à l'apprentissage automatique ?* reste ouverte. Cependant, ce chapitre nous a apporté plus d'informations et de recul sur cette question, qui devra encore être le centre de nombreuses études dans un futur proche.

Ainsi, l'étude des caractéristiques des solutions et des instances, permet de mieux comprendre les problèmes à résoudre, et la topologie des solutions, quelle que soit leur qualité. De nouvelles règles liant la qualité des solutions, leurs caractéristiques et les instances associées donnent le recul nécessaire pour imaginer de nouvelles méthodes, plus performantes, car adaptées à chaque instance.

Conclusion générale et perspectives

Synthèse des travaux

Dans le premier chapitre, nous avons étudié en détail les spécificités des problèmes de tournées, ainsi que les très nombreuses variantes associées. Un inventaire des méthodes de résolution, exactes et approchées, a été dressé pour le problème général, et deux variantes qui prennent de l'importance. L'étude des flottes hétérogènes a ainsi permis de prendre conscience de la complexité de cette variante, et du manque de spécificité des méthodes associées. Dans un second temps, nous avons pu mettre en avant l'arrivée d'un certain nombre de variantes adaptées à la mobilité urbaine. Enfin, ce chapitre met en lumière le temps nécessaire aux algorithmes compétitifs pour obtenir des solutions de bonne qualité, pour ainsi connaître les performances à atteindre.

Une fois l'état de l'art sur les avancées en recherche opérationnelle effectué, il devient nécessaire de se concentrer sur l'étude des méthodes d'analyse statistique. Le chapitre 2 consiste donc à étudier ce domaine de recherche et à voir les interactions existant avec l'optimisation combinatoire. Dans un premier temps, nous nous sommes concentrés sur les différentes branches de l'apprentissage automatique, et les méthodes associées. Ensuite, un état de l'art sur les méthodes impliquant les deux domaines d'étude a été effectué, mettant autant en lumière les méthodes améliorant en temps réel le solveur (apprentissage en ligne) que celles basées sur un historique de solution, récolté sur les résolutions d'instances plus anciennes (apprentissages hors ligne).

Le chapitre 3 détaille une première version d'un solveur, utilisant uniquement les connaissances en recherche opérationnelle. Celui-ci est basé sur le solveur de Soto *et al.* (2017) qui utilisait des voisinages basés sur le déplacement et l'inversion de portions de tournées, permettant ainsi de générer la majorité des voisinages de la littérature. Cette base nous a permis de générer et étudier un ensemble disjoint de voisinages avant d'en faire une implémentation optimisée et adaptée aux variantes à résoudre. Grâce à la réduction de graphe et à l'utilisation de la méthode SPLIT, le solveur a obtenu une réduction

significative du temps de résolution, tout en obtenant des performances intéressantes pour les problèmes hétérogènes. Enfin, ce solveur a été testé sur des instances avec un très grand nombre de clients, ainsi que sur des problèmes avec un nombre élevé de véhicules différents. Les solutions observées par RADOS sont de bonne qualité et obtenues très rapidement, malgré la complexité des problèmes testés, assurant ainsi une bonne base de résolution pour y ajouter des améliorations liées à l'apprentissage automatique.

Dans le chapitre 4, nous avons étudié les liens entre les caractéristiques d'une solution, celles de son instance associée, et sa qualité, afin de mieux comprendre ce qui permettra d'améliorer RADOS. Après avoir testé des méthodes simples et visuelles, nous avons dû nous concentrer sur des méthodes automatisables et plus complexes. C'est ainsi que nous nous sommes concentrés sur les travaux de Arnold et Sörensen (2019b) qui ont montré qu'il était possible de prédire la qualité d'une solution en observant uniquement certaines caractéristiques précises. Nous avons réussi à améliorer les connaissances sur le sujet en montrant l'absence d'impact des caractéristiques liées aux instances sur la performance globale des méthodes de classification. Nos recherches ont permis de montrer que l'étude en composantes principales permet d'identifier rapidement les caractéristiques les plus importantes, ainsi que la manière dont elles influent sur la qualité de la solution.

Enfin, nous avons étudié dans le chapitre 5 une nouvelle méthode, appelé *Feature Guided - Multiple Neighborhood Search* (FG-MNS) et combinant le solveur RADOS avec les connaissances acquises sur les caractéristiques des solutions. En développant un arbre de décision, nous avons pu déterminer des zones prometteuses dans l'espace des caractéristiques contenant majoritairement des solutions de bonne qualité, ainsi que des zones non prometteuses contenant peu de solutions intéressantes. Ces connaissances nous donnent l'opportunité de guider le solveur dans la recherche de bonnes solutions, notamment en déduisant les zones prometteuses grâce à l'observation des solutions issues d'instances déjà résolues. Cette nouvelle méthode a été testée sur un nouveau jeu d'instances créées à partir de données réelles. Afin d'améliorer l'apprentissage sur ces instances, celles-ci ont été divisées en classes, permettant ainsi de créer des règles adaptées aux spécificités de chaque groupe.

À l'issue de cette thèse, nous avons exploré les interactions possibles entre les méthodes d'analyse statistique et l'utilisation de métaheuristiques, appliquées à des problèmes de tournées issus de données réelles. Nous avons notamment implémenté un solveur adapté à de nombreuses variantes réelles, dont le temps d'exécution nécessaire pour obtenir des solutions intéressantes est inférieur à la littérature. Celui-ci a ensuite été modifié pour prendre en compte les connaissances acquises en étudiant les liens entre certaines caractéristiques des solutions, et leur coût. La méthode ainsi obtenue utilise les connaissances issues de problèmes déjà résolus, pour être guidée vers les zones prometteuses, contenant a priori les bonnes solutions. Cette cartographie des solutions est ensuite mise à jour pour prendre en compte les spécificités de l'instance à résoudre et améliorer ainsi la précision du guidage.

Limites et perspectives

Malgré de bonnes performances sur un jeu d'instances multipliant les difficultés (nombre de clients, faible temps de résolution, graphe non euclidien, flottes hétérogènes, etc.), la méthode FG-MNS ne parvient pas à améliorer sa version sans apprentissage. Celle-ci obtient en effet un excellent rapport qualité des solutions / temps d'exécution qui est difficile à améliorer. Cependant, ces travaux sont les premiers dans cette direction, et d'autres études devraient permettre prochainement d'aboutir à de meilleurs résultats.

En effet, plusieurs pistes sont prometteuses pour utiliser le meilleur de l'apprentissage automatique et de la fouille de données pour guider des solveur de recherche opérationnelle.

Par exemple, la faible performance des arbres pour l'apprentissage hors ligne semble indiquer que les groupes de solutions ne sont pas assez précis. Il est donc indispensable d'étudier de nouvelles caractéristiques définissant les instances, afin de mieux les partitionner.

De plus, le temps de calcul utilisé pour obtenir les solutions de la base d'entraînement de l'apprentissage hors ligne n'est pas un facteur limitant. Nous avons donc l'opportunité d'utiliser des algorithmes plus longs, mais plus performants, tels que l'UHGS (Vidal *et al.*, 2014), qui a également l'avantage d'imposer une plus grande diversité dans les solutions, ce qui améliorerait les connaissances sur la topologie des solutions.

Enfin, nous avons vu que les solutions de bonne qualité semblent dispersées en îlots de forte densité. Chacun de ces îlots pourrait donc être représenté par un nombre limité de solutions, ce qui permettrait de garder la même quantité d'informations avec moins de

solutions en entrée. La réduction du nombre de solutions nécessaires pour comprendre la topologie d'une instance permettra d'augmenter le nombre d'instances utilisées dans la base d'entraînement, améliorant encore un peu plus la robustesses des règles de décision associées.

L'expérience sur les arbres oracles montre qu'il y a également plusieurs points à améliorer concernant l'apprentissage en ligne. Des caractéristiques fondamentales n'ont probablement pas encore été découvertes et devront faire l'objet d'une étude poussée pour améliorer les performances des arbres de décision associés.

De plus, de nouveaux voisinages guidés devront voir le jour, pour réduire le risque d'atterrir dans un minimum local situé en dehors d'une zone prometteuse. L'étude de l'évolution des caractéristiques lors de l'utilisation de chaque voisinage permettra de choisir celui le plus efficace pour atteindre la zone prometteuse cible. Ces méthodes hybrides pourront également servir à orienter la réparation de solution et à l'optimisation de problème de VRP dynamique pour guider l'insertion de nouveaux clients. Enfin, le développement de méthodes constructives basées sur les caractéristiques pourrait permettre de créer une quantité importante de solutions de bonne qualité sans avoir à utiliser le guidage.

L'utilisation d'outils d'apprentissage automatique permet donc de développer de nouvelles manières de penser l'optimisation et ne demandent qu'à être étudiées davantage pour mettre en lumière les connaissances nécessaires pour créer les nouvelles méthodes à haute performance.

PUBLICATIONS

Revue avec acte

— Flavien Lucas, Romain Billot, and Marc Sevaux. A comment on "what makes a vrp solution good ? the generation of problem-specific knowledge for heuristics". *Computers & Operations Research*, 110 :130–134, 2019.

Conférence avec acte

— Flavien Lucas, Romain Billot, Marc Sevaux, and Kenneth Sörensen. Reducing space search in combinatorial optimization using machine learning tools. In : *International Conference on Learning and Intelligent Optimization*. Springer, Cham, 2020. p. 143-150.

Conférences sans acte

— Flavien Lucas, Romain Billot, and Marc Sevaux. Tournées de véhicules hétérogènes avec zones de circulation restreinte et trafic prédictif en milieu urbain. In *ROADEF : Recherche Opérationnelle et d'Aide à la Décision*, Lorient, France, February 2018.

— Flavien Lucas, Romain Billot, and Marc Sevaux. Amélioration de la résolution d'un problème de tournées de véhicules hétérogènes multi-attributs par des méthodes de machine learning. In *ROADEF : Recherche Opérationnelle et d'Aide à la Décision*, Le Havre, France, Février 2019.

— Flavien Lucas, Gwénaël Rault, Romain Billot, Philippe Lacomme, and Marc Sevaux. GRASP and Multiple Neighborhood Search for real vehicle routing problem. In *EURO 2019 - 30th European Conference on Operational Research*, Dublin, Ireland, June 2019.

— Gwénaël Rault, Flavien Lucas, and Marc Sevaux. Multiple solve approaches applied to the Heterogeneous Vehicle Routing Problem. In *Workshop of the EURO Working Group on Vehicle Routing and Logistics optimization (VeRoLog)*, page 158, Seville, Spain, June 2019.

— Flavien Lucas, Romain Billot, Marc Sevaux, and Kenneth Sörensen. Réduction de l'espace de recherche dans un MNS via l'utilisation de machine learning pour des problèmes de VRP. In *ROADEF 2020 : 21ème Congrès Annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Montpellier, France, February 2020.

Liste des Figures

LISTE DES TABLEAUX

BIBLIOGRAPHIE

Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, et Humaira Arshad. State-of-the-art in artificial neural network applications : A survey. *Heliyon*, 4(11) :e00938, 2018.

Alejandro Marcos Alvarez, Quentin Louveaux, et Louis Wehenkel. A supervised machine learning approach to variable branching in branch-and-bound. In *IN ECML*. Citeseer, 2014.

Christian Ambrosini et Jean-louis Routhier. Objectives, methods and results of surveys carried out in the field of urban freight transport : an international comparison. *Transport Reviews*, 24(1) :57–77, 2004.

Stephen Anderson, Julian Allen, et Michael Browne. Urban logistics—-how can it meet policy makers' sustainability objectives ? *Journal of transport geography*, 13(1) :71–81, 2005.

David Applegate et Robert Bixby. *Finding cuts in the TSP (A preliminary report)*. Citeseer, 1995.

David Applegate, Robert Bixby, William Cook, et Vasek Chvátal. On the solution of traveling salesman problems, 1998.

David Applegate, Robert Bixby, Vašek Chvátal, et William Cook. Tsp cuts which do not conform to the template paradigm. In *Computational combinatorial optimization*, pages 261–303. Springer, 2001.

David L Applegate, Robert E Bixby, Vasek Chvatal, et William J Cook. *The traveling salesman problem : a computational study*. Princeton university press, 2006.

Florian Arnold et Kenneth Sörensen. Knowledge-guided local search for the vehicle routing problem. *Computers & Operations Research*, 105 :32–46, 2019a.

Florian Arnold et Kenneth Sörensen. What makes a vrp solution good ? the generation of problem-specific knowledge for heuristics. *Computers & Operations Research*, 106 : 280–288, 2019b.

Florian Arnold, Michel Gendreau, et Kenneth Sörensen. Efficiently solving very large scale routing problems. Technical report, Cirrelt, 2017.

Florian Arnold, Michel Gendreau, et Kenneth Sörensen. Efficiently solving very large-scale routing problems. *Computers & Operations Research*, 107 :32–42, 2019a.

Florian Arnold, Ítalo Santana, Kenneth Sörensen, et Thibaut Vidal. Pils : Exploring high-order neighborhoods by pattern mining and injection. *arXiv preprint arXiv :1912.11462*, 2019b.

Roberto Baldacci, Maria Battarra, et Daniele Vigo. Routing a heterogeneous fleet of vehicles. In *The vehicle routing problem : latest advances and new challenges*, pages 3–27. Springer, 2008.

Roberto Baldacci, Paolo Toth, et Daniele Vigo. Exact algorithms for routing problems under vehicle capacity constraints. *Annals of Operations Research*, 175(1) :213–245, 2010.

Michel L Balinski et Richard E Quandt. On an integer program for a delivery problem. *Operations research*, 12(2) :300–304, 1964.

Cynthia Barnhart, Ellis L Johnson, George L Nemhauser, Martin WP Savelsbergh, et Pamela H Vance. Branch-and-price : Column generation for solving huge integer programs. *Operations research*, 46(3) :316–329, 1998.

Roberto Battiti et Giampietro Tecchiolli. The reactive tabu search. *ORSA journal on computing*, 6(2) :126–140, 1994.

John E Beasley. Route first—cluster second methods for vehicle routing. *Omega*, 11(4) : 403–408, 1983.

Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, et Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv :1611.09940*, 2016.

Yoshua Bengio, Andrea Lodi, et Antoine Prouvost. Machine learning for combinatorial optimization : a methodological tour d'horizon. *arXiv preprint arXiv :1811.06128*, 2018.

Gérard Biau et Erwan Scornet. A random forest guided tour. *Test*, 25(2) :197–227, 2016.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

Pierre Bonami, Andrea Lodi, et Giulia Zarpellon. Learning a classification of mixed-integer quadratic programming problems. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 595–604. Springer, 2018.

Léon Bottou, Frank E Curtis, et Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2) :223–311, 2018.

Justin Boyan et Andrew W Moore. Learning evaluation functions to improve optimization by local search. *Journal of Machine Learning Research*, 1(Nov) :77–112, 2000.

Kris Braekers, Katrien Ramaekers, et Inneke Van Nieuwenhuyse. The vehicle routing problem : State of the art classification and review. *Computers & Industrial Engineering*, 99 :300–313, 2016.

José Brandão. A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. *Computers & Operations Research*, 38(1) :140–151, 2011.

Mihaela Elena Breaban et Adrian Iftene. Dynamic objective sampling in many-objective optimization. *Procedia Computer Science*, 60 :178–187, 2015.

L Breiman, J. H Friedman, R. A Olshen, et C. J Stone. Classification and regression trees. *Cole Statistics/Probability Series*, 1984.

Leo Breiman. Random forests. *Machine learning*, 45(1) :5–32, 2001.

Bernd Bullnheimer, Richard F Hartl, et Christine Strauss. Applying the ant system to the vehicle routing problem. In *Meta-heuristics*, pages 285–296. Springer, 1999.

Rocsildes Canoy et Tias Guns. Vehicle routing by learning from historical solutions. In *International Conference on Principles and Practice of Constraint Programming*, pages 54–70. Springer, 2019.

Diego Cattaruzza, Nabil Absi, Dominique Feillet, et Jesús González-Feliu. Vehicle routing problems for city logistics. *EURO Journal on Transportation and Logistics*, 6(1) :51–79, 2017.

Alberto Ceselli, Giovanni Righini, et Matteo Salani. A column generation algorithm for a rich vehicle-routing problem. *Transportation Science*, 43(1) :56–69, 2009.

Eunjeong Choi et Dong-Wan Tcha. A column generation approach to the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, 34(7) :2080–2095, 2007.

Jan Christiaens et Greet Vanden Berghe. Slack induction by string removals for vehicle routing problems. *Transportation Science*, 54(2) :417–433, 2020.

Geoff Clarke et John W Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4) :568–581, 1964.

Jean-François Cordeau, Gilbert Laporte, et Anne Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society*, 52(8) :928–936, 2001.

Jean-François Cordeau, Gilbert Laporte, Martin WP Savelsbergh, et Daniele Vigo. Vehicle routing. *Handbooks in operations research and management science*, 14 :367–428, 2007.

David Corne, Clarisse Dhaenens, et Laetitia Jourdan. Synergies between operations research and data mining : The emerging use of multi-objective approaches. *European Journal of Operational Research*, 221(3) :469–479, 2012.

Teodor Gabriel Crainic, Nicoletta Ricciardi, et Giovanni Storchi. Advanced freight transportation systems for congested urban areas. *Transportation Research Part C : Emerging Technologies*, 12(2) :119–137, 2004.

Teodor Gabriel Crainic, Guido Perboli, Simona Mancini, et Roberto Tadei. Two-echelon vehicle routing problem : a satellite location analysis. *Procedia-Social and Behavioral Sciences*, 2(3) :5944–5955, 2010.

Rosario Cuda, Gianfranco Guastaroba, et Maria Grazia Speranza. A survey on two-echelon routing problems. *Computers & Operations Research*, 55 :185–199, 2015.

Paulo R de O da Costa, Jason Rhuggenaath, Yingqian Zhang, et Alp Akcay. Learning 2-opt heuristics for the traveling salesman problem via deep reinforcement learning. *arXiv preprint arXiv :2004.01608*, 2020.

George Dantzig, Ray Fulkerson, et Selmer Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4) :393–410, 1954.

George B Dantzig et John H Ramser. The truck dispatching problem. *Management science*, 6(1) :80–91, 1959.

Brian A Davis et Miguel A Figliozzi. A methodology to evaluate the competitiveness of electric delivery trucks. *Transportation Research Part E : Logistics and Transportation Review*, 49(1) :8–23, 2013.

Iman Dayarian, Teodor Gabriel Crainic, Michel Gendreau, et Walter Rei. An adaptive large-neighborhood search heuristic for a multi-period vehicle routing problem. *Transportation Research Part E : Logistics and Transportation Review*, 95 :95–123, 2016.

Martin Desrochers et TW Verhoog. A new heuristic for the fleet size and mix vehicle routing problem. *Computers & Operations Research*, 18(3) :263–274, 1991.

Marco Dorigo et Gianni Di Caro. Ant colony optimization : a new meta-heuristic. In *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1470–1477. IEEE, 1999.

Marco Dorigo, Mauro Birattari, et Thomas Stutzle. Ant colony optimization. *IEEE computational intelligence magazine*, 1(4) :28–39, 2006.

Christophe Duhamel, P Lacomme, et Caroline Prodhon. A graspxels with depth first search split procedure for the hvrp. *Res. Rep. LIMOS/RR-10-08 (Inst. Sup er. Inform., Mod el. Appl., Aubiere, France, 2010). Available at http ://www. isima. fr/lacomme/doc/RR_HVRP1-4_V1. pdf*, 2010.

Christophe Duhamel, Philippe Lacomme, et Caroline Prodhon. Efficient frameworks for greedy split and new depth first search split procedures for routing problems. *Computers & Operations Research*, 38(4) :723–739, 2011.

Jan Fabian Ehmke et Dirk Christian Mattfeld. Data allocation and application for time-dependent vehicle routing in city logistics. *European Transport*, pages 24–35, 2010.

Jan Fabian Ehmke, Stephan Meisel, et Dirk Christian Mattfeld. Floating car data based analysis of urban travel times for the provision of traffic quality. In *Traffic data collection and its standardization*, pages 129–149. Springer, 2010.

Jan Fabian Ehmke, André Steinert, et Dirk Christian Mattfeld. Advanced routing for city logistics service providers based on time-dependent travel times. *Journal of computational science*, 3(4) :193–205, 2012.

Sevgi Erdoğan et Elise Miller-Hooks. A green vehicle routing problem. *Transportation research part E : logistics and transportation review*, 48(1) :100–114, 2012.

Dominique Feillet. A tutorial on column generation and branch-and-price for vehicle routing problems. *4or*, 8(4) :407–424, 2010.

Thomas A Feo et Mauricio GC Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations research letters*, 8(2) :67–71, 1989.

Thomas A Feo et Mauricio GC Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2) :109–133, 1995.

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, et Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems ? *The journal of machine learning research*, 15(1) :3133–3181, 2014.

Marshall L Fisher. Optimal solution of vehicle routing problems using minimum k-trees. *Operations research*, 42(4) :626–642, 1994.

Marshall L Fisher et Ramchandran Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11(2) :109–124, 1981.

Anna Franceschetti, Dorothée Honhon, Gilbert Laporte, Tom Van Woensel, et Jan C Fransoo. Strategic fleet planning for city logistics. *Transportation Research Part B : Methodological*, 95 :19–40, 2017.

TJ Gaskell. Bases for vehicle fleet scheduling. *Journal of the Operational Research Society*, 18(3) :281–295, 1967.

Michel Gendreau, Alain Hertz, et Gilbert Laporte. A tabu search heuristic for the vehicle routing problem. *Management science*, 40(10) :1276–1290, 1994.

Michel Gendreau, Gilbert Laporte, Christophe Musaraganyi, et Éric D Taillard. A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, 26(12) :1153–1173, 1999.

Filip Gheysens, Bruce Golden, et Arjang Assad. A new heuristic for determining fleet size and composition. In *Netflow at Pisa*, pages 233–236. Springer, 1986.

Billy E Gillett et Leland R Miller. A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, 22(2) :340–349, 1974.

Paul C Gilmore et Ralph E Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9(6) :849–859, 1961.

Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers operations research*, 13(5) :533–549, 1986.

Fred Glover et Manuel Laguna. Tabu search. In *Handbook of combinatorial optimization*, pages 2093–2229. Springer, 1998.

David E Goldberg. *Genetic algorithms*. Pearson Education India, 2006.

Bruce Golden, Arjang Assad, Larry Levy, et Filip Gheysens. The fleet size and mix vehicle routing problem. *Computers & Operations Research*, 11(1) :49–66, 1984.

Bruce L Golden, Subramanian Raghavan, et Edward A Wasil. *The vehicle routing problem : latest advances and new challenges*, volume 43. Springer Science & Business Media, 2008.

Ian Goodfellow, Yoshua Bengio, et Aaron Courville. *Deep learning*. MIT press, 2016.

Abhijit Gosavi. Reinforcement learning : A tutorial survey and recent advances. *INFORMS Journal on Computing*, 21(2) :178–192, 2009.

Philippe Grangier, Michel Gendreau, Fabien Lehuédé, et Louis-Martin Rousseau. An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research*, 254 (1) :80–91, 2016.

Martin Grötschel. On the symmetric travelling salesman problem : solution of a 120-city problem. In *Combinatorial Optimization*, pages 61–77. Springer, 1980.

Astrid Gühnemann, Ralf-Peter Schäfer, Kai-Uwe Thiessenhusen, et Peter Wagner. Monitoring traffic and emissions by floating car data. *Institute of Transport Studies Working Paper*, 2004.

Andres E Gutierrez-Rodríguez, Santiago E Conant-Pablos, José C Ortiz-Bayliss, et Hugo Terashima-Marín. Selecting meta-heuristics for solving vehicle routing problems with time windows via meta-learning. *Expert Systems with Applications*, 118 :470–481, 2019.

J Pirie Hart et Andrew W Shogan. Semi-greedy heuristics : An empirical study. *Operations Research Letters*, 6(3) :107–114, 1987.

Keld Helsgaun. An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1) :106–130, 2000.

Keld Helsgaun. An extension of the lin-kernighan-helsgaun tsp solver for constrained traveling salesman and vehicle routing problems. *Roskilde : Roskilde University*, 2017.

JohnH Holland. adaptation in natural and artificial systems, university of michigan press, ann arbor,". *Cité page*, 100, 1975.

Stefan Irnich, Michael Schneider, et Daniele Vigo. Chapter 9 : Four variants of the vehicle routing problem. In *Vehicle Routing : Problems, Methods, and Applications, Second Edition*, pages 241–271. SIAM, 2014.

Jaber Jemai, Manel Zekri, et Khaled Mellouli. An nsga-ii algorithm for the green vehicle routing problem. In *European conference on evolutionary computation in combinatorial optimization*, pages 37–48. Springer, 2012.

David S Johnson et Lyle A McGeoch. The traveling salesman problem : A case study in local optimization. *Local search in combinatorial optimization*, 1(1) :215–310, 1997.

Chaitanya K Joshi, Thomas Laurent, et Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv preprint arXiv :1906.01227*, 2019.

Angel A Juan, Jarrod Goentzel, et Tolga Bektaş. Routing fleets with multiple driving ranges : Is it possible to use greener fleet configurations ? *Applied Soft Computing*, 21 : 84–94, 2014.

Jorge Kanda, Andre de Carvalho, Eduardo Hruschka, Carlos Soares, et Pavel Brazdil. Meta-learning to select the best meta-heuristic for the traveling salesman problem : A comparison of meta-features. *Neurocomputing*, 205 :393–406, 2016.

Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, et Le Song. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems*, pages 6348–6358, 2017a.

Elias B Khalil, Bistra Dilkina, George L Nemhauser, Shabbir Ahmed, et Yufen Shao. Learning to run heuristics in tree search. In *IJCAI*, pages 659–666, 2017b.

Elias Boutros Khalil, Pierre Le Bodic, Le Song, George Nemhauser, et Bistra Dilkina. Learning to branch in mixed integer programming. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

AP Kindervater et WP Savelsbergh. Chapter 10 in local search in combinatorial optimization, edited by e. aarts and jk lenstra, 1997.

Çağrı Koç, Tolga Bektaş, Ola Jabali, et Gilbert Laporte. Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research*, 249(1) :1–21, 2016.

Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

Lars Kotthoff. Llama : leveraging learning to automatically manage algorithms. *arXiv preprint arXiv :1306.1031*, 2013.

Lars Kotthoff. Algorithm selection for combinatorial search problems : A survey. In *Data Mining and Constraint Programming*, pages 149–190. Springer, 2016.

Lars Kotthoff, Ian P Gent, et Ian Miguel. An evaluation of machine learning in algorithm selection for search problems. *Ai Communications*, 25(3) :257–270, 2012.

Markus Kruber, Marco E Lübbecke, et Axel Parmentier. Learning when to use a decomposition. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 202–210. Springer, 2017.

Gilbert Laporte. The vehicle routing problem : An overview of exact and approximate algorithms. *European journal of operational research*, 59(3) :345–358, 1992.

Gilbert Laporte. Fifty years of vehicle routing. *Transportation science*, 43(4) :408–416, 2009.

Gilbert Laporte et Yves Nobert. A branch and bound algorithm for the capacitated vehicle routing problem. *Operations-Research-Spektrum*, 5(2) :77–85, 1983.

Feiyue Li, Bruce Golden, et Edward Wasil. A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, 34 (9) :2734–2742, 2007.

Canhong Lin, King Lun Choy, George TS Ho, Sai Ho Chung, et HY Lam. Survey of green vehicle routing problem : past and future trends. *Expert systems with applications*, 41 (4) :1118–1138, 2014.

Shen Lin et Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2) :498–516, 1973.

Fuh-Hwa Liu et Sheng-Yuan Shen. The fleet size and mix vehicle routing problem with time windows. *Journal of the Operational Research society*, 50(7) :721–732, 1999.

Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, et Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234 :11–26, 2017.

Andrea Lodi et Giulia Zarpellon. On learning and branching : a survey. *Top*, 25(2) : 207–236, 2017.

Alejandro Marcos Alvarez, Louis Wehenkel, et Quentin Louveaux. Online learning for strong branching approximation in branch-and-bound. Technical report, Université de Liège, 2016.

Renaud Masson, Fabien Lehuédé, et Olivier Péton. An adaptive large neighborhood search for the pickup and delivery problem with transfers. *Transportation Science*, 47 (3) :344–355, 2013.

Penélope Aguiar Melgarejo, Philippe Laborie, et Christine Solnon. A time-dependent no-overlap constraint : Application to urban delivery problems. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 1–17. Springer, 2015.

RH Mole et SR Jameson. A sequential route-building algorithm employing a generalised savings criterion. *Journal of the Operational Research Society*, 27(2) :503–511, 1976.

Lucien Mousin, Laetitia Jourdan, Marie-Eléonore Kessaci Marmion, et Clarisse Dhaenens. Feature selection using tabu search with learning memory : learning tabu search. In *International Conference on Learning and Intelligent Optimization*, pages 141–156. Springer, 2016.

Lucien Mousin, Marie-Eléonore Kessaci, et Clarisse Dhaenens. An iterated greedy-based approach exploiting promising sub-sequences of jobs to solve the no-wait flowshop scheduling problem. In *Metaheuristics International Conference*, 2017.

Abdelhamid Moutaoukil, Gilles Neubert, et Ridha Derrouiche. A comparison of homogeneous and heterogeneous vehicle fleet size in green vehicle routing problem. In *IFIP International Conference on Advances in Production Management Systems*, pages 450–457. Springer, 2014.

Jesús Muñuzuri, Rafael Grosso, Pablo Cortés, et José Guadix. Estimating the extra costs imposed on delivery vehicles using access time windows in a city. *Computers, Environment and Urban Systems*, 41 :262–275, 2013.

Kevin P Murphy. *Machine learning : a probabilistic perspective*. MIT press, 2012.

John C Nash. The (dantzig) simplex method for linear programming. *Computing in Science & Engineering*, 2(1) :29–31, 2000.

Mohammad Mahdi Nasiri, Sadegh Salesi, Ali Rahbari, Navid Salmanzadeh Meydani, et Mojtaba Abdollai. A data mining approach for population-based methods to solve the jssp. *Soft Computing*, 23(21) :11107–11122, 2019.

Gabriela Ochoa et Nadarajen Veerapen. Deconstructing the big valley search space hypothesis. In *Evolutionary Computation in Combinatorial Optimization*, pages 58–73. Springer, 2016.

Gabriela Ochoa et Nadarajen Veerapen. Mapping the global structure of tsp fitness landscapes. *Journal of Heuristics*, 24(3) :265–294, 2018.

Manfred Padberg et Giovanni Rinaldi. Facet identification for the symmetric traveling salesman polytope. *Mathematical programming*, 47(1-3) :219–257, 1990.

Diego Pecin, Artur Pessoa, Marcus Poggi, et Eduardo Uchoa. Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, 9(1) : 61–100, 2017.

Puca Huachi Vaz Penna, Anand Subramanian, Luiz Satoru Ochi, Thibaut Vidal, et Christian Prins. A hybrid heuristic for a broad class of vehicle routing problems with heterogeneous fleet. *Annals of Operations Research*, 273(1-2) :5–74, 2019.

Artur Pessoa, Eduardo Uchoa, et Marcus Poggi de Aragão. A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. *Networks : An International Journal*, 54(4) :167–177, 2009.

Artur Pessoa, Ruslan Sadykov, Eduardo Uchoa, et François Vanderbeck. A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, pages 1–41, 2020.

Victor Pillac, Michel Gendreau, Christelle Guéret, et Andrés L Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1) : 1–11, 2013.

Saikiran Pitla et al. Finding the kth largest item in a list of n items. *Department : Computer Science Indiana State University Terre Haute, IN, USA*, 2011.

Daniel Cosmin Porumbel, Jin-Kao Hao, et Pascale Kuntz. A search space "cartography" for guiding graph coloring heuristics. *Computers & Operations Research*, 37(4) :769–778, 2010.

Marcelo Prais et Celso C Ribeiro. Reactive grasp : An application to a matrix decomposition problem in tdma traffic assignment. *INFORMS Journal on Computing*, 12(3) : 164–176, 2000.

Christian Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12) :1985–2002, 2004.

Christian Prins. Two memetic algorithms for heterogeneous fleet vehicle routing problems. *Engineering Applications of Artificial Intelligence*, 22(6) :916–928, 2009.

Christian Prins, Philippe Lacomme, et Caroline Prodhon. Order-first split-second methods for vehicle routing problems : A review. *Transportation Research Part C : Emerging Technologies*, 40 :179–200, 2014.

Mingyao Qi, Wei-Hua Lin, Nan Li, et Lixin Miao. A spatiotemporal partitioning approach for large-scale vehicle routing problems with time windows. *Transportation Research Part E : Logistics and Transportation Review*, 48(1) :248–257, 2012.

J Ross Quinlan. *C4. 5 : programs for machine learning*. Elsevier, 2014.

Gwénaël Rault, Flavien Lucas, et Marc Sevaux. Multiple solve approaches applied to the heterogeneous vehicle routing problem. 2019.

Gerhard Reinelt. Tsplib—a traveling salesman problem library. *ORSA journal on computing*, 3(4) :376–384, 1991.

Jacques Renaud et Fayez F Boctor. A sweep-based algorithm for the fleet size and mix vehicle routing problem. *European Journal of Operational Research*, 140(3) :618–628, 2002.

Jacques Renaud, Fayez F Boctor, et Gilbert Laporte. An improved petal heuristic for the vehicle routeing problem. *Journal of the operational Research Society*, 47(2) :329–336, 1996.

Marcos Henrique Ribeiro, Alexandre Plastino, et Simone L Martins. Hybridization of grasp metaheuristic with data mining techniques. *Journal of Mathematical Modelling and Algorithms*, 5(1) :23–41, 2006.

John R Rice. The algorithm selection problem. In *Advances in computers*, volume 15, pages 65–118. Elsevier, 1976.

Ulrike Ritzinger, Jakob Puchinger, et Richard F Hartl. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research*, 54(1) : 215–231, 2016.

Stefan Ropke et David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4) : 455–472, 2006.

F Rosenblatt. The perceptron : A probabilistic model for information storage and organization in the brain1. *Psychological Review*, 65(6) :386–408, 1958.

Gavin A Rummery et Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

Ruslan Sadykov, Eduardo Uchoa, et Artur Pessoa. A bucket graph based labeling algorithm with application to vehicle routing. *Cadernos do LOGIS*, 7, 2017.

Michael Saint-Guillain, Christine Solnon, et Yves Deville. The static and stochastic vrp with time windows and both random customers and reveal times. In *European Conference on the Applications of Evolutionary Computation*, pages 110–127. Springer, 2017.

Said Salhi et Graham K Rand. Incorporating vehicle routing into the vehicle fleet composition problem. *European Journal of Operational Research*, 66(3) :313–330, 1993.

Dhish Kumar Saxena, João A Duro, Ashutosh Tiwari, Kalyanmoy Deb, et Qingfu Zhang. Objective reduction in many-objective optimization : Linear and nonlinear algorithms. *IEEE Transactions on Evolutionary Computation*, 17(1) :77–99, 2012.

Tommaso Schiavinotto et Thomas Stützle. A review of metrics on permutations for search landscape analysis. *Computers & operations research*, 34(10) :3143–3153, 2007.

Gerhard Schrimpf, Johannes Schneider, Hermann Stamm-Wilbrandt, et Gunter Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2) :139–171, 2000.

Maria Amélia Lopes Silva, Sérgio Ricardo de Souza, Marcone Jamilson Freitas Souza, et Ana Lúcia C Bazzan. A reinforcement learning-based multi-agent framework applied for solving routing and scheduling problems. *Expert Systems with Applications*, 131 : 148–171, 2019.

Robin A Skitt et Reuven R Levary. Vehicle routing via column generation. *European Journal of Operational Research*, 21(1) :65–76, 1985.

Elyn Solano-Charris, Christian Prins, et Andréa Cynthia Santos. Local search based metaheuristics for the robust vehicle routing problem with discrete scenarios. *Applied Soft Computing*, 32 :518–531, 2015.

Marius M Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2) :254–265, 1987.

Maria Soto, Marc Sevaux, André Rossi, et Andreas Reinholz. Multiple neighborhood search, tabu search and ejection chains for the multi-depot open vehicle routing problem. *Computers & Industrial Engineering*, 107 :211–222, 2017.

Suvrit Sra, Sebastian Nowozin, et Stephen J Wright. *Optimization for machine learning*. The MIT Press, 2012.

Anand Subramanian, Eduardo Uchoa, et Luiz Satoru Ochi. A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10) :2519–2531, 2013.

Richard S Sutton et Andrew G Barto. *Reinforcement learning : An introduction*. MIT press, 2018.

Éric D Taillard. A heuristic column generation method for the heterogeneous fleet vrp. *RAIRO-Operations Research-Recherche Opérationnelle*, 33(1) :1–14, 1999.

El-Ghazali Talbi. Combining metaheuristics with mathematical programming, constraint programming and machine learning. *Annals of Operations Research*, 240(1) :171–215, 2016.

Christos D Tarantilis et Chris T Kiranoudis. Boneroute : An adaptive memory-based method for effective fleet management. *Annals of operations Research*, 115(1-4) :227–241, 2002.

Túlio AM Toffolo, Thibaut Vidal, et Tony Wauters. Heuristics for vehicle routing problems : Sequence or set optimization ? *Computers & Operations Research*, 105 :118–131, 2019.

Paolo Toth et Daniele Vigo. The granular tabu search and its application to the vehicle-routing problem. *Informs Journal on computing*, 15(4) :333–346, 2003.

Paolo Toth et Daniele Vigo. *Vehicle routing : problems, methods, and applications*. SIAM, 2014.

Takahiro Tsubota, Ashish Bhaskar, Edward Chung, et Romain Billot. Arterial traffic congestion analysis using bluetooth duration data. In *Australasian Transport Research Forum*, 2011.

Renata Turkeš, Kenneth Sørensen, Lars Magnus Hvattum, Eva Barrena, Hayet Chentli, Leandro Coelho, Iman Dayarian, Axel Grimault, Anders Gullhav, Cagatay Iris, et al. Meta-analysis of metaheuristics : Quantifying the effect of adaptiveness in adaptive large neighborhood search. *University of Antwerp, Faculty of Business and Economics*, 2019.

Raras Tyasnurita, Ender Özcan, et Robert John. Learning heuristic selection using a time delay neural network for open vehicle routing. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1474–1481. IEEE, 2017.

Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, et Anand Subramanian. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3) :845–858, 2017.

Alex Van Breedam. *An Analysis of the Behavior of Heuristics for the Vehicle Routing Problem for a Selectrion of Problems with Vehicle-related, Customer-related, and Time-related Constraints*. RUCA, 1994.

Peter JM Van Laarhoven et Emile HL Aarts. Simulated annealing. In *Simulated annealing : Theory and applications*, pages 7–15. Springer, 1987.

François Vanderbeck et Laurence A Wolsey. An exact algorithm for ip column generation. *Operations research letters*, 19(4) :151–159, 1996.

Thibaut Vidal. Split algorithm in o (n) for the capacitated vehicle routing problem. *Computers & Operations Research*, 69 :40–47, 2016.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, Nadia Lahrichi, et Walter Rei. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60(3) :611–624, 2012.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, et Christian Prins. Heuristics for multi-attribute vehicle routing problems : A survey and synthesis. *European Journal of Operational Research*, 231(1) :1–21, 2013.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, et Christian Prins. A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, 234(3) :658–673, 2014.

Thibaut Vidal, Gilbert Laporte, et Piotr Matl. A concise guide to existing and emerging vehicle routing problem variants. *European Journal of Operational Research*, 2019.

Christos Voudouris et Edward Tsang. Guided local search and its application to the traveling salesman problem. *European journal of operational research*, 113(2) :469–499, 1999.

Christos Voudouris et Edward PK Tsang. Guided local search. In *Handbook of metaheuristics*, pages 185–218. Springer, 2003.

Christopher JCH Watkins et Peter Dayan. Q-learning. *Machine learning*, 8(3-4) :279–292, 1992.

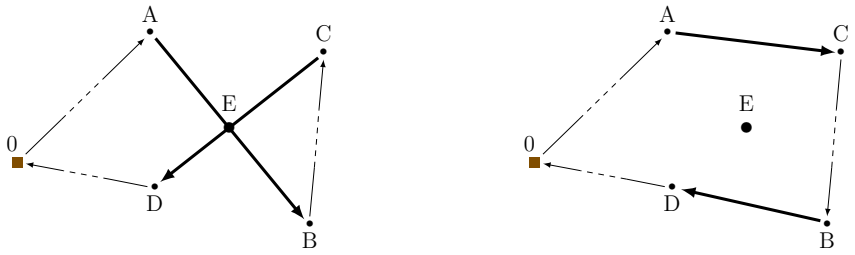
Hande Yaman. Formulations and valid inequalities for the heterogeneous vehicle routing problem. *Mathematical Programming*, 106(2) :365–390, 2006.

PC Yellow. A computational modification to the savings method of vehicle scheduling. *Journal of the Operational Research Society*, 21(2) :281–283, 1970.

ANNEXES

A.1 Non-optimalité des solutions du TSP avec croisements

Soient AB et CD deux arêtes d'une solution du TSP, se croisant en E.
Notons $C_0 := AB + CD$ le coût de ces deux arêtes. En réarrangeant les croisements (via un 2-opt, par exemple), on peut transformer les arêtes AB et CD en AC et BD, sans changer les autres arêtes du TSP. Notons $C_1 := AC + BD$, le coût de ces arêtes.



(a) Avec croisements (b) Sans croisements

FIGURE A1 – Graphes symétriques et asymétriques

Le segment AB peut se décomposer en $AE + EB$.
De plus, par symétrie, on obtient, $EB = BE$.
Ainsi, on a $AB = AE + BE$.
De façon similaire, on peut noter que $CD = EC + ED$.
Ainsi, $C_0 = AE + BE + EC + ED$.
Ce qui peut être réagencé en : $C_0 = AE + EC + BE + ED$.

L'inégalité triangulaire nous indique que $AE + EC \geq AC$ et $BE + ED \geq BD$.

On obtient finalement : $C_0 \geq AC + BD$; donc : $\boxed{C_0 \geq C_1}$

Dans un problème de TSP avec des distances euclidiennes, le coût d'une solution sans croisement est inférieur ou égal au coût d'une solution avec croisements. Parmi les solutions optimales, il y en a forcément une qui n'a pas de croisements.

A.2 Pseudo-code de l'algorithme SPLIT

L'algorithme SPLIT est une partie fondamentale des solveurs développés pendant la thèse, en améliorant l'affectation clients/véhicules (voir chapitre 3 pour plus détails). Cette méthode prends en entrée une permutation, représentant un ordre précis de visite des clients. Dans une première phase, l'algorithme identifie le découpage à coût minimal respectant chacune des contraintes (algorithme 1). Cette solution étant écrit sous forme d'une suite d'étiquettes de indiquant les retours au dépôt, elle est convertie au format standard en appliquant l'algorithme 2.

Algorithme 1 : SPLIT

```
1  pour i de 1 à n faire V_i = +∞ fin;
2  pour i de 1 à n faire
3      charge := 0 ; coût := 0 ; j := i;
4      répéter
5          charge := charge + q_{S_j};
6          si i = j alors
7              coût := c_{0,S_j} + d_{S_j} + c_{s_j,0}
8          sinon
9              coût := coût − c_{S_{j−1},0} + c_{S_{j−1},S_j} + d_{S_j} + c_{S_j,0}
10         si (charge ≤ W) et (coût ≤ L) alors
11             // Ici S_i ... S_j correspondent aux arc (i − 1, j) dans H si
                V_{i−1}+coût< V_j alors
12                 V_j := V_{i−1}+coût;
13                 P_j := i − 1;
14             j := j + 1;
15     jusqu'à (j > n) ou (charge > W) ou (coût > L);
```

Algorithme 2 : décodage du SPLIT

```
1  pour i de 1 à n faire
2      trip(i) := ∅
3  t := 0 ; j := n;
4  répéter
5      t := t + 1;
6      i := P_j;
7      pour k de i + 1 à j faire
8          enfiler(trip(t), S_k)
9  jusqu'à i = 0;
```

A.3 Astuce de réduction de graphe

Nous avons vu au chapitre 3 l'importance de réduire la taille du graphe. Pour obtenir un tel graphe, la méthode la plus simple consiste à trier pour chaque client i les distances $c_{i,j}$ et de choisir les k plus petites valeurs. Cependant, cette méthode est en $O(n\ \log_2(n))$ pour chaque client, donc en $O(n^2 \log_2(n))$ au total, ce qui entraîne un temps d'exécution conséquent pour les problèmes comportant plusieurs milliers de clients.

Afin de gagner du temps CPU, nous avons cherché des méthodes permettant de connaître le k-eme plus petit élément d'une liste. Ce problème est assez peu étudié, et le seul algorithme testé (Pitla et al., 2011) ne nous a pas satisfait en terme de performances.

Nous avons donc créé notre propre méthode.

Soit S une suite d'éléments de taille n dont on voudrais connaître et trier les k plus petits éléments. Prenons par exemple une liste à réduire de taille $n = 20$ et $k = 5$, illustrée dans la figure A2.

77	3	37	49	75	38	36	79	85	1	94	71	55	96	79	3	18	33	35	79

FIGURE A2 – Exemple de liste à réduire et trier

L'idée principale consiste à trouver dans un temps très court une borne supérieur de la valeur du k-eme élément. Ainsi, tout élément supérieur à cette valeur pivot est certain de ne pas faire partie des k éléments à conserver et peut donc être éliminé.

Pour cela, nous divisons la liste d'éléments en k sous-listes de taille similaire. La plus petite valeur de chaque cluster est ensuite extraite, et on identifie la plus grande de ces valeurs. Notons p cette valeur.

Dans notre exemple (figure A3), les plus petites valeurs sont respectivement $3, 36, 1, 3$ et 18, dont la plus grande valeur p est **36**

77	**3**	37	49	75	38	**36**	79	85	**1**	94	71	55	96	79	**3**	**18**	33	35	79

FIGURE A3 – Valeurs minimale dans chaque cluster

Nous savons donc que p est minorée par les $k-1$ autre valeurs minimales des clusters. Nous venons donc d'obtenir une valeur supérieur ou égale au k eme plus petit élément de notre liste : p est notre pivot.

Les valeurs supérieures à p peuvent donc être éliminées (figure A4).

77	3	37	49	75	38	**36**	79	85	1	94	71	55	96	79	3	18	33	35	79

FIGURE A4 – Élimination des valeurs supérieures au pivot

Cette méthode a donc éliminé des solutions peu intéressantes en parcourant deux fois la liste des éléments (une première fois pour trouver la valeur du pivot, puis une deuxième fois pour comparer les éléments au pivot), elle est donc de complexité en $O(n)$.

Ensuite, il suffit juste de trier les éléments restant et de ne conserver que les k plus petits (figure A5).

1	3	3	18	33	35	36

FIGURE A5 – Exemple de liste à réduire et trier

Si toutes les grandes valeurs se trouvent dans le même cluster, le pivot sera de taille conséquente, et n'éliminera que $\lfloor \frac{n}{k} \rfloor - 1$ solutions. Cependant, en pratique le gain est vraiment conséquent : sur les instances XXL, le temps de réduction de graphe, sans l'utilisation de pivot varie de 1 min pour l'instance de 3000 clients à plus de 15 min pour l'instance à 10 000 clients (les instances plus grandes n'ont pas été testées sans le pivot) là où le pivot permet de réduire ce temps à quelques secondes.

A.4 Détail de la résolution des instances DLP

A.4.1 Groupe A

Tableau A1 – Résultats détaillés sur les instances DLP (groupe A)

Nom	N	C	Objectif	GAP (%)	Temps CPU
HVRP_DLP_01	92,00	4,00	9597,61	4,21	1,23
HVRP_DLP_02	181,00	4,00	12941,90	11,09	6,52
HVRP_DLP_03	124,00	4,00	11762,20	9,83	4,14
HVRP_DLP_04	183,00	4,00	11731,80	9,49	9,05
HVRP_DLP_05	116,00	5,00	12125,10	11,56	4,21
HVRP_DLP_06	121,00	8,00	13099,90	12,13	3,96
HVRP_DLP_07	108,00	4,00	8626,23	6,87	2,23
HVRP_DLP_08	84,00	3,00	4791,14	4,34	0,59
HVRP_DLP_09	167,00	5,00	8210,71	8,04	3,31
HVRP_DLP_10	69,00	4,00	2197,48	4,27	0,33
HVRP_DLP_11	95,00	4,00	3461,63	2,80	1,15
HVRP_DLP_12	112,00	4,00	3589,20	1,28	0,93
HVRP_DLP_17	105,00	3,00	5724,02	6,75	1,17
HVRP_DLP_21	126,00	3,00	5318,43	3,49	1,53
HVRP_DLP_26	126,00	5,00	6944,84	8,41	2,23
HVRP_DLP_29	164,00	4,00	9440,16	3,37	3,67
HVRP_DLP_2A	113,00	6,00	8335,12	6,95	4,91
HVRP_DLP_30	112,00	3,00	6429,57	2,41	0,92
HVRP_DLP_31	131,00	8,00	4289,01	4,84	6,65
HVRP_DLP_34	136,00	6,00	6046,56	5,36	4,04
HVRP_DLP_35	168,00	6,00	10429,50	9,53	6,26
HVRP_DLP_36	85,00	6,00	6202,73	9,13	2,62
HVRP_DLP_40	132,00	5,00	12070,70	9,18	3,40
HVRP_DLP_43	86,00	7,00	9046,50	3,54	2,04
HVRP_DLP_47	111,00	5,00	17030,70	5,41	2,48
HVRP_DLP_48	111,00	5,00	22651,40	6,56	2,25
HVRP_DLP_51	129,00	3,00	8370,88	8,42	2,29

Tableau A1 (suite) - Résultats détaillés sur les instances DLP (groupe A)

Nom	N	C	Objectif	GAP (%)	Temps CPU
HVRP_DLP_52	59,00	3,00	4140,77	2,83	0,23
HVRP_DLP_53	115,00	3,00	7024,62	9,18	1,65
HVRP_DLP_55	56,00	3,00	10554,00	3,02	0,18
HVRP_DLP_56	153,00	4,00	32678,90	5,74	2,38
HVRP_DLP_60	137,00	4,00	17750,70	4,34	1,76
HVRP_DLP_63	174,00	5,00	22306,40	12,14	4,52
HVRP_DLP_67	172,00	5,00	11972,80	11,14	8,79
HVRP_DLP_68	125,00	4,00	9861,47	10,94	3,16
HVRP_DLP_69	152,00	4,00	10132,40	11,01	3,68
HVRP_DLP_71	186,00	3,00	10424,60	6,39	2,28
HVRP_DLP_72	186,00	4,00	6554,17	11,65	4,57
HVRP_DLP_73	137,00	5,00	10570,30	3,68	3,76
HVRP_DLP_74	125,00	5,00	12099,90	4,43	2,49
HVRP_DLP_75	20,00	3,00	456,75	0,86	0,03
HVRP_DLP_79	147,00	4,00	7545,10	3,96	2,53
HVRP_DLP_80	171,00	3,00	7159,31	5,02	3,08
HVRP_DLP_81	106,00	4,00	10945,10	3,42	1,28
HVRP_DLP_82	79,00	3,00	4879,25	3,41	0,64
HVRP_DLP_83	124,00	4,00	10582,00	5,80	1,39
HVRP_DLP_84	105,00	4,00	7562,84	4,63	1,39
HVRP_DLP_85	146,00	4,00	9260,86	5,67	1,88
HVRP_DLP_86	153,00	5,00	9480,82	5,10	2,89
HVRP_DLP_87	108,00	4,00	3956,39	5,40	1,31
HVRP_DLP_88	127,00	5,00	14371,70	16,03	2,38
HVRP_DLP_90	102,00	4,00	2413,73	5,96	0,93
HVRP_DLP_92	35,00	3,00	627,15	11,12	0,05
HVRP_DLP_93	40,00	3,00	1065,61	2,76	0,19
HVRP_DLP_94	47,00	5,00	1461,82	6,06	0,22

A.4.2 Groupe B

Tableau A2 – Résultats détaillés sur les instances DLP (groupe B)

Nom	N	C	Objectif	GAP (%)	Temps CPU
HVRP_DLP_13	119,00	5,00	7138,35	6,60	3,79
HVRP_DLP_14	176,00	4,00	5922,61	5,01	2,81
HVRP_DLP_15	188,00	7,00	9080,40	10,45	5,47
HVRP_DLP_16	129,00	6,00	4240,96	2,04	3,17
HVRP_DLP_18	256,00	5,00	10120,40	4,89	6,19
HVRP_DLP_19	224,00	5,00	12699,10	8,67	8,22
HVRP_DLP_22	239,00	2,00	13822,60	5,92	0,68
HVRP_DLP_23	203,00	4,00	8147,93	5,24	4,16
HVRP_DLP_24	163,00	4,00	9939,72	9,20	2,69
HVRP_DLP_25	143,00	6,00	7673,47	6,47	4,04
HVRP_DLP_27	220,00	5,00	8955,26	6,32	5,51
HVRP_DLP_28	141,00	5,00	5678,07	2,77	4,75
HVRP_DLP_2B	107,00	6,00	9065,83	7,13	4,30
HVRP_DLP_32	244,00	8,00	10619,80	13,19	14,42
HVRP_DLP_33	189,00	7,00	10112,80	8,36	7,44
HVRP_DLP_37	161,00	5,00	7341,10	7,36	4,76
HVRP_DLP_38	205,00	5,00	12039,20	7,55	7,57
HVRP_DLP_39	77,00	5,00	3074,59	5,37	1,51
HVRP_DLP_41	135,00	7,00	8745,48	15,41	6,04
HVRP_DLP_42	178,00	7,00	12399,40	14,76	14,23
HVRP_DLP_44	172,00	3,00	13579,30	11,38	2,26
HVRP_DLP_45	170,00	3,00	11799,60	12,63	4,01
HVRP_DLP_46	250,00	5,00	26457,40	7,70	8,92
HVRP_DLP_49	246,00	8,00	17930,90	10,81	28,42
HVRP_DLP_50	187,00	6,00	12993,90	5,04	5,83
HVRP_DLP_54	172,00	4,00	11368,10	9,83	8,62
HVRP_DLP_57	163,00	4,00	47508,50	6,09	2,86
HVRP_DLP_58	220,00	6,00	25790,40	10,36	6,10
HVRP_DLP_59	193,00	6,00	15328,00	7,32	8,16

Tableau A2 (suite) - Résultats détaillés sur les instances DLP (groupe B)

Nom	N	C	Objectif	GAP (%)	Temps CPU
HVRP_DLP_61	111,00	3,00	7710,34	5,74	1,00
HVRP_DLP_62	225,00	5,00	25268,60	9,93	6,68
HVRP_DLP_64	161,00	3,00	17615,90	2,81	1,57
HVRP_DLP_65	223,00	3,00	14031,30	7,57	4,84
HVRP_DLP_66	150,00	4,00	13730,80	7,47	3,89
HVRP_DLP_70	78,00	4,00	7109,23	6,35	1,69
HVRP_DLP_76	152,00	8,00	13548,50	12,96	3,96
HVRP_DLP_77	190,00	3,00	7366,38	6,50	1,84
HVRP_DLP_78	190,00	4,00	7407,76	5,28	3,06
HVRP_DLP_89	134,00	5,00	7735,63	9,15	4,38
HVRP_DLP_91	196,00	4,00	6868,75	8,20	2,84
HVRP_DLP_95	184,00	2,00	6376,08	3,34	0,79

A.5 Détail de la résolution des instances réelles

A.5.1 Flotte homogène

Tableau A3 – Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Avignon	193	ikea	commerces	0,47	0,11	0,31	0,80	0,00	0,03	0,59
Avignon	250	ikea	bar	1,04	0,00	1,07	0,40	0,40	0,62	0,69
Avignon	250	ikea	arrêts de bus	0,06	0,00	0,80	0,28	0,28	0,56	0,53
Avignon	193	railway	commerces	0,85	0,00	1,42	1,27	1,13	0,23	1,53
Avignon	250	railway	bar	1,04	0,00	0,54	1,05	1,61	1,30	1,38
Avignon	250	railway	arrêts de bus	0,62	0,00	1,25	0,92	0,92	0,02	1,29
Avignon	193	ikea	commerces	0,92	0,03	1,41	0,00	1,01	0,23	0,62
Avignon	300	ikea	bar	0,00	0,23	0,77	0,70	0,51	0,62	1,88
Avignon	300	ikea	arrêts de bus	0,26	0,00	0,80	0,90	1,48	0,55	0,66
Avignon	193	railway	commerces	1,02	0,19	1,19	0,74	0,00	0,87	0,75
Avignon	300	railway	bar	1,26	0,00	1,16	1,40	1,40	0,54	1,23
Avignon	300	railway	arrêts de bus	0,79	0,00	0,55	0,71	0,84	0,66	0,55
Avignon	193	ikea	commerces	0,30	0,00	0,64	0,47	0,27	0,42	0,20
Avignon	350	ikea	bar	0,32	0,00	1,40	0,96	0,96	1,02	2,06
Avignon	350	ikea	arrêts de bus	0,53	0,00	0,76	0,47	0,56	0,18	0,58
Avignon	193	railway	commerces	1,23	0,00	0,96	1,93	0,18	0,08	1,47
Avignon	350	railway	bar	1,03	0,52	0,00	2,18	3,78	0,98	2,92
Avignon	350	railway	arrêts de bus	0,76	0,00	0,39	0,82	1,13	0,82	0,73
Avignon	193	ikea	commerces	0,52	0,00	0,71	0,81	1,11	1,15	0,94
Avignon	400	ikea	bar	0,00	0,32	0,17	1,14	1,22	0,33	0,33
Avignon	400	ikea	arrêts de bus	0,67	0,00	1,02	0,98	0,57	0,73	0,94
Avignon	193	railway	commerces	0,50	0,00	0,78	0,03	0,34	0,77	0,18
Avignon	400	railway	bar	1,09	0,00	1,26	1,10	1,28	0,26	0,26
Avignon	399	railway	arrêts de bus	0,42	0,00	0,30	1,29	1,05	0,90	0,97
Avignon	193	ikea	commerces	0,49	0,00	0,61	1,57	1,31	0,66	0,71
Avignon	450	ikea	bar	0,31	0,00	0,49	0,34	0,34	0,34	0,34
Avignon	449	ikea	arrêts de bus	2,15	0,00	3,12	2,74	3,05	2,65	2,65
Avignon	193	railway	commerces	0,18	0,25	0,40	0,70	0,70	0,68	0,00
Avignon	450	railway	bar	0,00	0,30	1,78	0,98	1,54	1,05	1,81
Avignon	449	railway	arrêts de bus	1,52	0,78	0,00	1,23	1,62	1,48	1,54
Avignon	193	ikea	commerces	0,37	0,00	0,94	0,75	0,86	0,73	0,20
Avignon	464	ikea	bar	1,85	0,00	1,65	2,00	0,35	1,64	1,78
Avignon	500	ikea	arrêts de bus	1,29	0,00	1,07	1,11	1,19	1,47	1,55
Avignon	193	railway	commerces	0,46	0,00	0,60	0,75	0,96	0,59	0,71
Avignon	464	railway	bar	0,71	0,00	2,53	1,99	2,67	1,08	0,42
Avignon	499	railway	arrêts de bus	0,86	0,00	1,26	0,83	0,91	0,80	0,93
Bordeaux	250	ikea	commerces	0,73	0,00	1,58	0,64	0,77	2,33	0,82
Bordeaux	248	ikea	bar	3,16	0,00	2,21	2,29	2,74	2,42	2,23
Bordeaux	247	ikea	arrêts de bus	0,42	0,00	1,17	0,21	1,65	0,81	0,74
Bordeaux	250	railway	commerces	1,39	0,00	1,04	1,34	1,36	0,32	0,73
Bordeaux	248	railway	bar	0,49	0,00	1,43	0,81	0,81	0,66	0,35
Bordeaux	250	railway	arrêts de bus	1,14	0,00	0,89	1,58	1,43	1,27	1,73
Bordeaux	300	ikea	commerces	1,25	0,00	0,69	2,22	1,64	1,70	2,96
Bordeaux	298	ikea	bar	2,17	0,00	1,20	1,80	1,39	2,00	1,86
Bordeaux	299	ikea	arrêts de bus	0,18	0,00	0,71	2,93	0,81	2,18	1,48
Bordeaux	300	railway	commerces	0,89	0,00	0,82	1,81	1,81	1,62	0,24
Bordeaux	298	railway	bar	0,17	0,00	1,99	2,52	2,00	2,26	1,46
Bordeaux	298	railway	arrêts de bus	2,26	0,00	2,97	2,64	3,08	2,14	2,14
Bordeaux	350	ikea	commerces	2,15	0,00	3,27	3,37	4,80	3,26	1,32
Bordeaux	347	ikea	bar	1,24	0,00	1,85	1,28	2,93	1,50	1,50
Bordeaux	347	ikea	arrêts de bus	2,16	0,00	1,88	3,19	1,90	2,89	3,11
Bordeaux	350	railway	commerces	0,97	0,00	1,54	2,95	1,57	1,11	1,50
Bordeaux	348	railway	bar	1,25	0,00	2,32	2,23	1,16	1,30	1,67
Bordeaux	349	railway	arrêts de bus	1,01	0,00	1,39	2,99	1,84	1,76	1,71
Bordeaux	400	ikea	commerces	1,14	0,00	1,46	3,50	2,21	1,29	3,38
Bordeaux	397	ikea	bar	1,12	0,00	1,03	1,77	0,74	1,22	0,29

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Bordeaux	398	ikea	arrêts de bus	0,49	0,00	0,85	1,87	1,87	1,39	1,39
Bordeaux	400	railway	commerces	1,68	0,00	2,48	3,08	2,95	2,49	2,84
Bordeaux	397	railway	bar	1,24	0,00	0,30	1,21	0,88	0,28	0,28
Bordeaux	395	railway	arrêts de bus	2,68	0,00	1,62	3,04	3,63	2,89	1,76
Bordeaux	450	ikea	commerces	0,68	0,00	2,23	3,11	4,09	2,13	2,13
Bordeaux	447	ikea	bar	1,72	0,00	1,08	1,07	2,08	1,10	2,06
Bordeaux	447	ikea	arrêts de bus	1,02	0,00	2,01	2,95	3,80	1,84	1,84
Bordeaux	450	railway	commerces	1,16	0,00	0,74	1,78	0,96	1,16	1,22
Bordeaux	448	railway	bar	1,50	0,00	2,24	2,22	2,33	1,65	1,20
Bordeaux	445	railway	arrêts de bus	1,25	0,40	0,00	1,44	0,78	1,34	1,53
Bordeaux	500	ikea	commerces	1,17	0,00	2,43	0,98	0,42	1,26	1,26
Bordeaux	492	ikea	bar	1,60	0,00	2,63	1,43	1,84	2,47	2,46
Bordeaux	496	ikea	arrêts de bus	0,78	0,00	2,07	1,50	1,50	2,31	1,98
Bordeaux	500	railway	commerces	0,30	0,00	1,61	1,11	1,91	0,36	1,68
Bordeaux	496	railway	bar	1,72	0,00	3,48	2,74	3,58	2,01	2,16
Bordeaux	498	railway	arrêts de bus	1,72	0,00	1,98	1,14	1,14	1,06	2,76
Brest	248	ikea	arrêts de bus	1,49	0,00	1,31	1,63	1,61	1,58	1,37
Brest	249	railway	arrêts de bus	0,64	0,00	1,07	0,26	0,77	0,67	0,67
Brest	299	ikea	arrêts de bus	0,46	0,18	2,14	0,48	0,00	0,48	0,48
Brest	300	railway	arrêts de bus	0,68	0,00	0,30	0,55	0,62	0,87	1,01
Brest	350	railway	arrêts de bus	1,51	0,00	2,42	1,82	1,82	1,79	2,43
Brest	399	ikea	arrêts de bus	1,52	0,00	1,74	1,21	1,33	1,21	1,35
Brest	450	ikea	arrêts de bus	0,45	0,00	1,04	0,14	0,14	0,14	1,35
Brest	450	railway	arrêts de bus	0,38	0,00	0,65	0,38	0,38	0,34	0,25
Brest	500	ikea	arrêts de bus	0,33	0,00	0,57	0,38	0,38	0,38	0,38
Clermont-Ferrand	250	ikea	commerces	0,93	0,00	0,58	1,43	0,79	1,30	0,67
Clermont-Ferrand	249	ikea	bar	0,00	0,02	0,31	1,18	1,46	1,78	1,87
Clermont-Ferrand	250	ikea	arrêts de bus	0,52	0,00	0,39	1,58	0,93	2,10	0,71
Clermont-Ferrand	250	railway	commerces	0,07	0,02	0,00	0,73	0,49	0,16	0,05
Clermont-Ferrand	250	railway	bar	0,31	0,31	0,76	0,00	0,80	0,43	1,05
Clermont-Ferrand	250	railway	arrêts de bus	0,47	0,00	2,17	2,26	1,24	0,53	1,35
Clermont-Ferrand	297	ikea	commerces	0,80	0,00	0,92	0,84	0,74	0,54	0,54
Clermont-Ferrand	300	ikea	bar	0,63	0,00	1,40	1,25	1,25	0,12	0,79
Clermont-Ferrand	300	ikea	arrêts de bus	0,00	0,22	1,13	2,16	1,56	0,94	1,21
Clermont-Ferrand	297	railway	commerces	0,60	0,00	0,69	1,70	0,55	0,81	0,81
Clermont-Ferrand	300	railway	bar	0,21	0,10	0,96	0,53	0,26	0,00	0,54
Clermont-Ferrand	300	railway	arrêts de bus	0,63	0,15	1,29	1,57	0,00	1,01	1,07
Clermont-Ferrand	297	ikea	commerces	1,07	0,00	1,72	1,02	1,65	1,28	1,62
Clermont-Ferrand	349	ikea	bar	0,53	0,00	1,58	0,83	0,79	1,80	2,39
Clermont-Ferrand	350	ikea	arrêts de bus	1,05	0,00	1,50	1,61	1,61	1,10	1,61
Clermont-Ferrand	297	railway	commerces	0,00	0,28	1,34	1,42	0,84	0,99	0,99
Clermont-Ferrand	350	railway	bar	0,19	0,00	1,16	0,58	1,09	0,87	1,25
Clermont-Ferrand	350	railway	arrêts de bus	0,31	0,00	0,90	1,43	0,13	0,49	1,33
Clermont-Ferrand	297	ikea	commerces	0,14	0,26	1,43	1,84	2,49	0,00	0,12
Clermont-Ferrand	400	ikea	bar	0,62	0,00	1,11	1,15	1,15	1,81	1,74
Clermont-Ferrand	400	ikea	arrêts de bus	0,09	0,00	1,23	0,62	1,38	1,09	1,74
Clermont-Ferrand	297	railway	commerces	0,39	0,00	0,44	2,04	0,87	0,23	0,52
Clermont-Ferrand	399	railway	bar	0,82	0,12	1,00	1,11	0,70	0,00	0,70
Clermont-Ferrand	400	railway	arrêts de bus	0,38	0,00	0,93	1,23	1,49	0,63	0,49
Clermont-Ferrand	297	ikea	commerces	0,72	0,00	1,90	1,85	1,56	1,90	0,14
Clermont-Ferrand	449	ikea	bar	0,54	0,00	0,74	0,55	0,55	0,85	0,85
Clermont-Ferrand	450	ikea	arrêts de bus	0,71	0,00	1,89	1,12	1,58	1,21	1,39
Clermont-Ferrand	297	railway	commerces	0,23	0,00	0,72	1,57	0,44	0,44	0,44
Clermont-Ferrand	449	railway	bar	0,44	0,00	2,05	1,24	1,49	0,90	1,46
Clermont-Ferrand	450	railway	arrêts de bus	1,18	0,00	0,67	1,59	1,39	0,48	0,71
Clermont-Ferrand	297	ikea	commerces	1,02	0,23	0,07	0,00	1,04	1,29	2,04
Clermont-Ferrand	499	ikea	bar	1,37	0,00	1,29	1,60	2,14	1,26	1,56
Clermont-Ferrand	500	ikea	arrêts de bus	0,00	0,14	1,09	1,18	1,81	1,07	1,07
Clermont-Ferrand	297	railway	commerces	0,35	0,00	0,55	1,35	0,70	0,66	0,85
Clermont-Ferrand	499	railway	bar	0,66	0,00	1,18	1,10	0,13	1,02	1,02
Clermont-Ferrand	500	railway	arrêts de bus	0,26	0,00	0,67	0,00	1,10	0,45	0,45
Dijon	250	ikea	commerces	0,41	0,00	0,75	0,27	0,87	0,51	0,94
Dijon	250	ikea	bar	1,30	0,65	1,22	0,00	2,03	0,12	2,39
Dijon	250	ikea	arrêts de bus	3,23	0,00	3,59	0,22	1,84	1,68	2,33

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Dijon	250	railway	commerces	0,71	0,00	1,01	1,25	0,59	0,77	0,60
Dijon	250	railway	bar	1,28	0,00	1,68	0,63	2,27	1,23	1,53
Dijon	250	railway	arrêts de bus	0,70	0,00	0,55	1,96	1,39	1,59	1,10
Dijon	271	ikea	commerces	1,01	0,00	0,50	1,33	1,47	0,82	0,82
Dijon	300	ikea	bar	0,39	0,00	0,49	0,44	0,44	0,26	0,89
Dijon	299	ikea	arrêts de bus	0,77	0,00	1,60	1,54	1,54	1,12	1,56
Dijon	271	railway	commerces	0,45	0,00	1,98	0,68	0,61	2,36	2,36
Dijon	299	railway	bar	0,13	0,00	1,26	1,06	1,79	0,48	1,45
Dijon	300	railway	arrêts de bus	0,26	0,03	0,93	0,00	0,51	0,22	0,40
Dijon	271	ikea	commerces	0,29	0,00	0,63	0,23	0,67	0,33	0,69
Dijon	349	ikea	bar	0,58	0,00	1,50	1,42	1,56	1,37	1,12
Dijon	350	ikea	arrêts de bus	0,62	0,00	1,02	2,64	2,43	0,98	0,98
Dijon	271	railway	commerces	0,28	0,17	0,78	0,71	0,00	0,36	0,72
Dijon	349	railway	bar	0,14	0,00	0,05	0,64	0,73	1,22	1,82
Dijon	349	railway	arrêts de bus	0,00	0,36	1,01	1,32	1,99	0,83	1,19
Dijon	271	ikea	commerces	0,19	0,00	0,19	0,12	0,12	0,09	0,09
Dijon	399	ikea	bar	1,97	0,00	1,32	1,45	2,27	1,80	2,54
Dijon	399	ikea	arrêts de bus	1,21	0,00	1,09	1,64	1,57	0,79	1,18
Dijon	271	railway	commerces	0,25	0,00	0,17	0,44	0,44	0,19	0,35
Dijon	399	railway	bar	1,13	0,00	1,00	1,10	1,11	1,37	2,31
Dijon	400	railway	arrêts de bus	0,28	0,00	0,57	0,02	0,02	0,49	0,38
Dijon	271	ikea	commerces	0,56	0,00	0,89	1,11	1,43	0,75	0,98
Dijon	449	ikea	bar	1,07	0,00	0,74	1,02	1,02	0,94	0,94
Dijon	449	ikea	arrêts de bus	0,43	0,00	1,00	0,46	0,64	0,98	1,07
Dijon	271	railway	commerces	0,05	0,00	0,46	0,74	0,64	0,89	0,32
Dijon	449	railway	bar	0,91	0,33	1,57	1,64	2,27	0,00	0,00
Dijon	449	railway	arrêts de bus	1,11	0,00	1,45	1,68	1,55	0,59	1,56
Dijon	271	ikea	commerces	1,32	0,23	1,01	0,00	1,26	0,39	1,10
Dijon	470	ikea	bar	0,72	0,00	1,10	0,53	0,53	0,78	1,06
Dijon	500	ikea	arrêts de bus	1,25	0,00	2,80	1,94	1,53	2,47	2,51
Dijon	271	railway	commerces	0,42	0,17	0,15	0,82	1,24	0,00	0,93
Dijon	470	railway	bar	4,39	2,48	3,54	3,89	0,00	2,79	2,46
Dijon	500	railway	arrêts de bus	1,30	0,00	1,84	0,38	2,78	1,49	2,88
Grenoble	250	ikea	commerces	0,77	0,00	0,79	1,05	1,72	0,84	0,88
Grenoble	250	ikea	bar	0,39	0,00	0,59	1,34	0,87	0,49	0,60
Grenoble	250	ikea	arrêts de bus	0,00	0,04	1,09	1,05	0,84	0,06	1,01
Grenoble	250	railway	commerces	0,18	0,00	0,20	0,43	0,61	0,59	0,77
Grenoble	250	railway	bar	0,33	0,00	0,36	0,34	0,34	0,47	0,68
Grenoble	250	railway	arrêts de bus	0,29	0,00	0,92	0,51	0,59	0,19	0,62
Grenoble	300	ikea	commerces	1,24	0,52	1,39	1,22	1,76	0,00	1,37
Grenoble	300	ikea	bar	0,58	0,00	0,64	0,26	0,39	0,12	0,12
Grenoble	300	ikea	arrêts de bus	0,68	0,00	1,53	1,79	0,99	0,47	2,11
Grenoble	300	railway	commerces	0,20	0,00	0,06	0,28	0,52	0,43	0,46
Grenoble	300	railway	bar	0,25	0,00	0,24	0,23	0,44	0,36	0,30
Grenoble	299	railway	arrêts de bus	0,33	0,00	0,63	0,58	0,85	0,55	0,39
Grenoble	350	ikea	commerces	0,55	0,00	0,92	3,24	0,61	0,75	0,51
Grenoble	350	ikea	bar	0,00	0,02	0,74	0,50	1,81	1,40	1,28
Grenoble	350	ikea	arrêts de bus	0,75	0,00	0,63	7,34	0,49	3,77	0,49
Grenoble	350	railway	commerces	0,68	0,07	0,32	0,25	0,82	0,27	0,00
Grenoble	350	railway	bar	0,25	0,09	0,47	0,36	0,00	0,24	0,29
Grenoble	350	railway	arrêts de bus	0,39	0,00	0,61	0,43	0,42	0,32	0,41
Grenoble	400	ikea	commerces	0,93	0,00	1,16	3,97	1,11	3,65	2,70
Grenoble	399	ikea	bar	0,04	0,00	0,04	1,49	0,63	1,50	1,92
Grenoble	400	ikea	arrêts de bus	0,90	0,00	1,24	6,18	1,47	2,67	1,62
Grenoble	400	railway	commerces	0,26	0,00	1,93	0,08	0,88	0,65	0,59
Grenoble	400	railway	bar	0,13	0,00	0,18	0,30	0,08	0,26	0,26
Grenoble	400	railway	arrêts de bus	0,51	0,00	1,35	0,49	1,12	0,70	0,84
Grenoble	450	ikea	commerces	1,30	0,00	0,64	9,76	0,96	1,19	0,85
Grenoble	449	ikea	bar	0,94	0,00	0,68	1,45	0,66	0,54	0,54
Grenoble	450	ikea	arrêts de bus	0,71	0,00	1,39	2,91	1,34	1,14	1,63
Grenoble	450	railway	commerces	0,47	0,00	0,45	0,60	0,60	0,60	0,49
Grenoble	449	railway	bar	0,40	0,00	0,51	0,04	0,58	0,34	0,48
Grenoble	450	railway	arrêts de bus	0,74	0,00	1,11	0,84	0,42	0,36	0,90
Grenoble	500	ikea	commerces	0,08	0,00	0,41	2,79	2,19	0,10	3,69

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Grenoble	500	ikea	bar	0,41	0,00	0,61	2,07	0,69	1,37	0,73
Grenoble	499	ikea	arrêts de bus	0,08	0,15	0,89	0,35	0,47	0,00	0,35
Grenoble	500	railway	commerces	0,20	0,01	0,00	0,47	0,62	0,35	0,47
Grenoble	500	railway	bar	0,37	0,00	0,34	0,11	0,45	0,61	0,39
Grenoble	500	railway	arrêts de bus	0,78	0,00	0,64	0,68	0,68	0,71	0,99
Lyon	250	ikea	commerces	0,22	0,00	0,30	1,32	0,46	0,45	0,16
Lyon	247	ikea	bar	0,82	0,00	0,58	0,44	0,44	1,30	1,28
Lyon	244	ikea	arrêts de bus	0,80	0,29	0,71	0,69	0,69	0,00	1,72
Lyon	250	railway	commerces	0,19	0,00	0,49	0,51	0,62	0,39	0,80
Lyon	247	railway	bar	0,70	0,00	0,66	0,51	1,65	1,20	1,30
Lyon	248	railway	arrêts de bus	0,03	0,00	0,70	0,71	0,70	0,57	0,88
Lyon	300	ikea	commerces	0,19	0,00	1,24	1,10	1,10	0,88	0,83
Lyon	297	ikea	bar	0,81	0,04	0,84	0,83	1,04	1,10	0,00
Lyon	297	ikea	arrêts de bus	0,99	0,00	1,65	2,01	1,57	0,83	1,79
Lyon	300	railway	commerces	0,48	0,00	0,40	0,61	0,76	0,50	0,56
Lyon	297	railway	bar	0,21	0,00	0,62	0,43	0,86	0,70	1,22
Lyon	296	railway	arrêts de bus	0,39	0,00	0,75	0,52	0,18	0,52	0,83
Lyon	350	ikea	commerces	0,72	0,20	0,96	0,00	0,83	1,11	0,53
Lyon	347	ikea	bar	1,73	0,39	0,00	1,95	2,45	1,01	2,08
Lyon	345	ikea	arrêts de bus	0,87	0,00	0,78	1,50	1,20	0,62	0,53
Lyon	350	railway	commerces	0,71	0,00	0,19	0,49	1,01	0,49	0,75
Lyon	347	railway	bar	1,18	0,00	1,30	1,65	1,04	1,30	1,01
Lyon	345	railway	arrêts de bus	0,67	0,00	0,69	0,51	0,61	0,51	0,51
Lyon	400	ikea	commerces	1,42	0,00	1,87	1,64	2,67	1,62	2,08
Lyon	395	ikea	bar	0,52	0,00	1,09	2,19	0,91	2,64	1,62
Lyon	397	ikea	arrêts de bus	0,00	0,29	0,33	0,66	1,39	0,73	0,98
Lyon	400	railway	commerces	0,31	0,00	0,38	0,58	0,31	0,65	0,85
Lyon	396	railway	bar	0,15	0,00	0,74	1,05	1,51	1,54	0,66
Lyon	392	railway	arrêts de bus	0,21	0,00	0,52	0,97	0,21	1,40	1,18
Lyon	450	ikea	commerces	1,24	0,00	0,40	2,83	1,80	2,29	2,31
Lyon	446	ikea	bar	0,70	0,00	1,39	0,93	1,31	1,16	1,04
Lyon	445	ikea	arrêts de bus	0,48	0,00	1,44	2,30	1,63	0,76	1,15
Lyon	450	railway	commerces	0,75	0,00	1,73	1,21	0,90	0,53	1,90
Lyon	445	railway	bar	1,33	1,14	2,60	0,00	0,00	1,33	1,78
Lyon	443	railway	arrêts de bus	0,39	0,00	1,24	0,58	1,06	0,49	0,73
Lyon	500	ikea	commerces	0,39	0,00	1,61	0,99	1,05	0,20	1,10
Lyon	495	ikea	bar	0,32	0,00	1,05	1,78	1,10	1,23	1,50
Lyon	490	ikea	arrêts de bus	0,87	0,00	0,26	0,81	0,92	0,81	1,47
Lyon	500	railway	commerces	0,79	0,00	0,57	0,65	0,93	0,65	1,04
Lyon	495	railway	bar	1,24	0,00	1,70	0,82	1,45	0,84	1,18
Lyon	489	railway	arrêts de bus	0,26	0,00	0,05	0,66	0,36	0,24	0,71
Marseille	250	ikea	commerces	1,80	0,95	1,42	0,00	1,29	1,70	1,05
Marseille	250	ikea	bar	0,63	0,00	0,47	0,79	0,17	0,66	0,66
Marseille	250	ikea	arrêts de bus	0,99	0,00	2,29	1,73	1,82	1,08	1,89
Marseille	250	railway	commerces	0,99	0,00	1,47	1,22	1,85	1,87	2,04
Marseille	250	railway	bar	0,65	0,00	0,76	0,51	0,61	0,86	0,62
Marseille	250	railway	arrêts de bus	1,09	0,00	0,90	1,10	1,54	1,10	1,30
Marseille	300	ikea	commerces	0,53	0,00	1,02	1,23	0,57	1,23	0,79
Marseille	300	ikea	bar	0,83	0,00	1,30	1,29	1,30	1,31	1,38
Marseille	300	ikea	arrêts de bus	0,97	0,00	1,01	0,89	0,89	0,71	1,56
Marseille	300	railway	commerces	1,09	0,00	1,00	0,67	0,89	0,99	1,14
Marseille	300	railway	bar	0,24	0,00	0,90	0,91	0,96	0,28	0,28
Marseille	300	railway	arrêts de bus	0,25	0,00	0,78	1,05	0,74	1,17	1,30
Marseille	350	ikea	commerces	0,50	0,00	0,71	0,45	1,15	0,51	1,36
Marseille	350	ikea	bar	1,51	0,00	0,84	1,20	1,43	1,35	1,29
Marseille	350	ikea	arrêts de bus	1,49	0,00	1,22	1,80	1,71	1,62	1,71
Marseille	350	railway	commerces	1,73	0,00	1,71	1,48	2,35	2,19	1,60
Marseille	350	railway	bar	0,39	0,37	0,45	0,82	0,82	0,82	0,00
Marseille	350	railway	arrêts de bus	0,00	0,26	0,92	1,35	1,16	1,05	1,55
Marseille	400	ikea	commerces	1,67	0,00	3,22	2,02	1,62	2,28	2,49
Marseille	400	ikea	bar	0,68	0,00	0,16	0,57	0,74	1,05	0,99
Marseille	400	ikea	arrêts de bus	0,61	0,00	0,79	0,26	0,48	1,00	0,38
Marseille	400	railway	commerces	0,60	0,00	0,38	1,02	0,97	0,32	0,93
Marseille	400	railway	bar	0,33	0,00	0,80	1,01	1,01	0,71	1,47

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Marseille	400	railway	arrêts de bus	0,59	0,00	0,53	0,47	0,97	0,53	1,11
Marseille	450	ikea	commerces	1,21	0,00	0,67	1,29	1,05	1,02	2,03
Marseille	450	ikea	bar	0,88	0,00	1,52	1,49	1,97	1,77	1,77
Marseille	450	ikea	arrêts de bus	0,84	0,00	1,81	1,60	1,27	1,12	1,65
Marseille	450	railway	commerces	1,38	0,00	1,87	1,08	1,28	1,28	1,23
Marseille	450	railway	bar	0,00	0,10	1,11	1,57	0,11	0,39	2,45
Marseille	450	railway	arrêts de bus	0,71	0,00	1,92	1,12	1,68	1,30	1,32
Marseille	500	ikea	commerces	0,82	0,00	1,66	0,90	1,29	0,41	1,11
Marseille	500	ikea	bar	0,10	0,00	0,39	1,33	0,98	0,58	1,33
Marseille	500	ikea	arrêts de bus	0,38	0,00	0,82	0,24	1,42	0,24	1,29
Marseille	500	railway	commerces	0,76	0,00	1,23	0,53	0,53	0,53	0,53
Marseille	500	railway	bar	0,40	0,00	1,65	1,18	0,09	0,35	0,79
Marseille	500	railway	arrêts de bus	0,57	0,14	0,50	0,00	1,10	0,60	0,65
Metz	218	ikea	commerces	0,31	0,00	1,00	0,51	1,21	1,32	1,32
Metz	250	ikea	bar	1,92	0,01	1,02	0,00	0,15	0,42	0,15
Metz	250	ikea	arrêts de bus	0,71	0,00	1,36	0,44	0,44	0,93	1,17
Metz	218	railway	commerces	0,60	0,00	0,74	0,84	0,95	0,52	0,36
Metz	249	railway	bar	0,61	0,00	1,15	0,59	0,48	0,59	0,85
Metz	250	railway	arrêts de bus	0,79	0,00	1,00	0,81	2,60	1,48	1,28
Metz	218	ikea	commerces	0,89	0,00	1,36	2,87	1,74	0,00	0,57
Metz	299	ikea	bar	0,06	0,00	3,12	2,79	1,45	1,86	0,76
Metz	300	ikea	arrêts de bus	1,36	0,00	1,66	2,07	1,79	1,61	1,69
Metz	218	railway	commerces	0,33	0,00	0,91	0,85	1,47	0,44	1,13
Metz	299	railway	bar	0,97	0,47	0,00	0,84	1,37	1,22	1,50
Metz	300	railway	arrêts de bus	0,46	0,00	0,36	0,62	0,87	0,58	0,74
Metz	218	ikea	commerces	0,00	0,02	1,27	0,41	1,78	1,21	1,87
Metz	348	ikea	bar	0,32	0,00	1,68	1,70	1,06	1,67	1,62
Metz	350	ikea	arrêts de bus	0,97	0,00	1,17	2,78	0,33	1,42	1,16
Metz	218	railway	commerces	0,48	0,20	0,68	0,00	0,75	0,93	0,49
Metz	349	railway	bar	0,69	0,10	0,00	1,41	1,54	0,55	1,45
Metz	350	railway	arrêts de bus	0,66	0,00	0,78	1,15	1,25	1,15	0,66
Metz	218	ikea	commerces	0,38	0,00	0,68	3,56	0,78	1,84	2,16
Metz	398	ikea	bar	0,36	0,00	1,89	1,45	2,96	1,51	2,06
Metz	400	ikea	arrêts de bus	0,83	0,14	1,42	0,68	1,39	0,37	0,00
Metz	218	railway	commerces	0,44	0,07	0,26	0,00	0,12	0,27	0,36
Metz	398	railway	bar	0,55	0,04	0,61	0,46	1,21	0,00	0,00
Metz	400	railway	arrêts de bus	1,03	0,00	1,19	1,04	1,50	0,89	1,08
Metz	218	ikea	commerces	1,37	0,00	0,56	1,01	3,07	1,82	0,33
Metz	414	ikea	bar	0,95	0,00	1,29	1,18	3,17	1,55	1,69
Metz	450	ikea	arrêts de bus	1,14	0,00	1,40	1,50	1,83	1,49	1,49
Metz	218	railway	commerces	0,53	0,00	0,70	1,12	0,72	0,56	1,45
Metz	414	railway	bar	1,08	0,00	1,51	1,50	1,50	1,71	1,71
Metz	450	railway	arrêts de bus	0,30	0,00	1,26	0,09	0,44	1,07	0,07
Metz	218	ikea	commerces	0,00	0,00	1,89	4,87	1,44	1,99	2,99
Metz	414	ikea	bar	0,76	0,00	0,68	1,78	0,90	0,31	1,24
Metz	500	ikea	arrêts de bus	1,08	0,00	1,81	1,20	1,02	1,89	1,34
Metz	218	railway	commerces	0,62	0,00	1,12	1,00	1,00	0,80	1,77
Metz	414	railway	bar	1,37	0,00	0,63	1,74	0,62	1,20	1,23
Metz	500	railway	arrêts de bus	1,17	0,00	2,29	2,81	2,76	1,41	2,02
Montpellier	250	ikea	commerces	0,94	0,00	2,07	0,55	1,08	0,34	0,95
Montpellier	250	ikea	bar	0,78	0,00	0,66	0,86	1,21	1,27	1,50
Montpellier	250	ikea	arrêts de bus	0,09	0,00	0,32	0,57	0,66	0,31	0,12
Montpellier	250	railway	commerces	0,27	0,00	0,46	0,12	0,43	0,83	0,31
Montpellier	250	railway	bar	1,82	0,75	2,47	0,00	0,00	0,00	0,00
Montpellier	250	railway	arrêts de bus	0,37	0,00	0,79	0,53	0,04	2,09	0,88
Montpellier	300	ikea	commerces	0,93	0,00	1,12	2,27	1,67	1,32	2,99
Montpellier	300	ikea	bar	0,22	0,00	0,26	2,01	1,46	0,79	0,64
Montpellier	300	ikea	arrêts de bus	0,09	0,00	0,97	0,98	0,98	0,27	0,59
Montpellier	300	railway	commerces	0,57	0,00	1,05	0,92	0,95	2,37	0,76
Montpellier	300	railway	bar	0,28	0,00	0,60	0,38	0,33	0,14	0,14
Montpellier	300	railway	arrêts de bus	0,32	0,17	0,00	0,39	0,61	0,21	0,74
Montpellier	350	ikea	commerces	1,10	0,00	1,45	1,42	1,41	1,24	1,61
Montpellier	350	ikea	bar	1,18	0,00	1,90	0,54	1,46	1,17	1,16
Montpellier	350	ikea	arrêts de bus	0,89	0,00	0,33	1,37	1,70	1,08	1,08

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Montpellier	350	railway	commerces	0,20	0,26	0,54	0,00	0,00	1,40	0,68
Montpellier	350	railway	bar	0,92	0,15	0,00	0,40	1,29	0,40	0,51
Montpellier	350	railway	arrêts de bus	0,90	0,00	0,86	0,99	0,99	0,67	1,04
Montpellier	400	ikea	commerces	0,64	0,00	1,14	1,10	1,88	0,28	1,10
Montpellier	400	ikea	bar	0,59	0,00	0,60	2,41	2,21	1,85	1,91
Montpellier	400	ikea	arrêts de bus	0,42	0,00	0,37	0,30	0,30	0,30	0,30
Montpellier	400	railway	commerces	0,78	0,00	0,16	0,68	0,30	0,43	0,30
Montpellier	400	railway	bar	0,52	0,37	0,57	0,69	0,58	0,00	0,41
Montpellier	400	railway	arrêts de bus	0,41	0,00	1,05	0,91	0,83	0,92	0,48
Montpellier	450	ikea	commerces	0,78	0,00	0,77	0,83	0,94	0,90	0,90
Montpellier	450	ikea	bar	0,62	0,00	0,36	0,87	0,11	0,93	2,78
Montpellier	450	ikea	arrêts de bus	1,34	0,00	1,81	1,49	2,47	1,56	1,95
Montpellier	450	railway	commerces	1,12	0,00	1,51	0,86	1,02	0,86	1,38
Montpellier	450	railway	bar	0,13	0,15	0,78	0,00	0,00	0,00	0,00
Montpellier	450	railway	arrêts de bus	0,29	0,00	0,61	0,33	0,33	0,33	0,33
Montpellier	500	ikea	commerces	0,79	0,00	0,37	0,60	0,82	0,79	0,86
Montpellier	500	ikea	bar	1,19	0,00	1,65	1,45	1,00	1,02	1,06
Montpellier	500	ikea	arrêts de bus	0,26	0,00	0,44	0,48	0,98	0,27	1,31
Montpellier	500	railway	commerces	0,31	0,00	1,06	0,40	0,25	0,40	0,40
Montpellier	500	railway	bar	0,72	0,00	0,83	1,01	1,35	1,01	1,19
Montpellier	500	railway	arrêts de bus	0,26	0,00	0,31	0,50	0,50	0,82	0,54
Nantes	250	ikea	commerces	1,05	0,00	2,15	3,34	1,72	2,10	1,50
Nantes	249	ikea	bar	0,29	0,00	0,31	0,39	0,63	0,38	0,31
Nantes	250	ikea	arrêts de bus	0,67	0,00	1,05	1,76	1,85	0,96	2,12
Nantes	250	railway	commerces	0,87	0,00	1,47	1,49	1,62	1,89	1,62
Nantes	249	railway	bar	1,51	0,00	0,26	1,44	1,20	1,61	1,36
Nantes	249	railway	arrêts de bus	0,84	0,00	1,04	0,58	1,38	1,33	1,42
Nantes	300	ikea	commerces	0,77	0,00	0,78	0,57	0,52	0,32	0,50
Nantes	299	ikea	bar	0,75	0,24	1,23	1,04	0,49	0,00	0,18
Nantes	300	ikea	arrêts de bus	1,03	0,00	1,84	1,77	1,68	0,57	2,10
Nantes	300	railway	commerces	0,01	0,00	0,84	0,35	1,11	1,82	0,30
Nantes	299	railway	bar	1,59	0,00	1,50	1,35	1,35	1,79	0,49
Nantes	300	railway	arrêts de bus	0,71	0,00	0,71	0,88	0,88	0,76	0,82
Nantes	350	ikea	commerces	2,47	0,00	2,89	2,41	2,85	1,99	2,75
Nantes	349	ikea	bar	0,68	0,00	0,62	0,38	0,56	1,17	0,20
Nantes	349	ikea	arrêts de bus	1,00	0,00	1,53	1,50	2,87	0,82	1,15
Nantes	350	railway	commerces	0,88	0,00	1,31	0,76	0,96	0,98	1,13
Nantes	349	railway	bar	0,25	0,00	0,11	0,27	0,59	0,60	0,16
Nantes	350	railway	arrêts de bus	0,74	0,00	0,69	1,10	1,08	0,47	0,48
Nantes	400	ikea	commerces	1,54	0,00	1,46	1,59	1,73	1,36	1,36
Nantes	399	ikea	bar	0,00	0,10	0,14	0,71	0,74	0,80	0,73
Nantes	400	ikea	arrêts de bus	0,81	0,00	0,83	1,26	1,67	0,93	0,93
Nantes	400	railway	commerces	1,08	0,00	0,25	0,97	1,41	0,97	1,70
Nantes	399	railway	bar	1,00	0,00	0,91	1,37	1,48	1,12	0,93
Nantes	400	railway	arrêts de bus	1,14	0,00	0,94	1,26	1,03	0,98	0,91
Nantes	450	ikea	commerces	0,73	0,00	1,81	0,94	1,53	0,99	0,99
Nantes	449	ikea	bar	0,38	0,00	1,27	0,59	0,70	0,25	1,00
Nantes	450	ikea	arrêts de bus	0,99	0,00	1,30	1,53	1,34	1,37	1,21
Nantes	450	railway	commerces	0,00	0,10	0,60	0,09	0,09	0,09	0,71
Nantes	449	railway	bar	0,58	0,00	0,44	1,08	0,97	0,44	0,70
Nantes	449	railway	arrêts de bus	1,40	0,00	1,02	1,06	0,89	1,21	1,22
Nantes	500	ikea	commerces	1,45	0,00	2,07	1,58	1,58	1,44	1,44
Nantes	499	ikea	bar	0,55	0,00	0,38	0,65	0,78	0,12	0,58
Nantes	500	ikea	arrêts de bus	0,76	0,00	1,60	1,53	1,87	0,81	1,45
Nantes	500	railway	commerces	0,47	0,00	0,72	0,64	0,62	1,12	0,71
Nantes	499	railway	bar	0,58	0,00	0,44	0,52	0,57	0,06	0,27
Nantes	500	railway	arrêts de bus	0,69	0,00	0,62	0,88	1,08	0,65	0,96
Orléans	250	ikea	commerces	0,48	0,00	0,81	0,63	0,63	0,50	0,94
Orléans	250	ikea	bar	0,29	0,00	1,08	0,62	1,20	1,00	2,39
Orléans	250	ikea	arrêts de bus	1,01	0,00	1,55	1,51	1,28	1,06	1,46
Orléans	250	railway	commerces	0,64	0,00	0,86	1,26	1,26	0,96	0,80
Orléans	250	railway	bar	0,21	0,00	0,20	1,11	0,51	0,72	0,96
Orléans	250	railway	arrêts de bus	0,99	0,00	1,10	0,78	0,78	0,78	1,20
Orléans	300	ikea	commerces	0,88	0,75	1,18	2,32	1,37	0,00	0,37

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Orléans	299	ikea	bar	0,42	0,00	0,54	0,92	0,09	0,60	0,37
Orléans	300	ikea	arrêts de bus	0,56	0,00	0,96	1,11	1,45	1,05	1,05
Orléans	300	railway	commerces	1,14	0,00	1,51	1,86	0,87	3,09	1,36
Orléans	299	railway	bar	0,49	0,00	0,42	0,92	1,06	0,79	0,91
Orléans	300	railway	arrêts de bus	1,38	0,00	1,96	0,87	2,13	2,66	1,23
Orléans	330	ikea	commerces	0,67	0,00	1,92	1,96	1,25	0,40	0,40
Orléans	349	ikea	bar	0,64	0,00	0,16	0,83	1,01	1,14	0,90
Orléans	350	ikea	arrêts de bus	0,22	0,00	2,18	2,75	1,27	1,50	1,92
Orléans	330	railway	commerces	0,73	0,00	0,70	0,26	0,05	0,95	0,87
Orléans	350	railway	bar	0,93	0,00	1,08	0,29	0,33	0,96	0,74
Orléans	350	railway	arrêts de bus	0,64	0,00	0,65	0,68	0,68	0,43	0,94
Orléans	330	ikea	commerces	0,75	0,00	1,08	2,45	0,83	1,22	1,69
Orléans	399	ikea	bar	0,67	0,00	1,24	1,73	1,73	1,81	0,75
Orléans	400	ikea	arrêts de bus	1,03	0,74	1,59	1,53	2,40	0,00	1,98
Orléans	330	railway	commerces	0,38	0,00	1,21	1,25	0,74	2,01	1,70
Orléans	399	railway	bar	0,74	0,00	0,71	0,26	0,64	0,91	0,91
Orléans	400	railway	arrêts de bus	1,31	0,00	1,80	1,71	1,67	1,34	1,93
Orléans	330	ikea	commerces	0,76	0,00	1,22	0,90	0,90	0,42	1,48
Orléans	450	ikea	bar	0,42	0,00	0,52	2,47	1,14	0,97	1,79
Orléans	450	ikea	arrêts de bus	0,47	0,00	0,42	0,24	0,99	0,40	0,80
Orléans	330	railway	commerces	0,25	0,00	0,77	0,87	0,87	2,69	1,49
Orléans	449	railway	bar	0,39	0,00	0,28	0,62	0,68	0,62	0,80
Orléans	450	railway	arrêts de bus	1,20	0,00	1,98	1,44	1,90	1,01	1,01
Orléans	330	ikea	commerces	0,55	0,15	0,88	0,39	0,39	0,00	1,04
Orléans	499	ikea	bar	1,43	0,00	1,85	1,70	1,70	1,23	1,23
Orléans	500	ikea	arrêts de bus	0,60	0,08	1,27	0,29	0,62	0,00	0,86
Orléans	330	railway	commerces	0,57	0,00	0,56	0,42	0,92	0,30	0,30
Orléans	499	railway	bar	0,43	0,34	1,15	0,97	0,73	1,06	0,00
Orléans	500	railway	arrêts de bus	0,78	0,00	3,40	1,05	1,06	1,05	1,04
Reims	243	ikea	commerces	0,77	0,00	0,83	0,43	0,91	0,40	0,40
Reims	250	ikea	bar	1,37	0,00	2,33	0,64	1,79	2,44	2,54
Reims	250	ikea	arrêts de bus	0,87	0,00	1,20	1,18	1,18	0,98	0,88
Reims	243	railway	commerces	0,71	0,00	0,51	0,64	0,64	0,51	0,21
Reims	250	railway	bar	0,10	0,00	1,29	0,91	0,51	0,66	1,20
Reims	250	railway	arrêts de bus	0,28	0,00	2,22	1,96	2,38	0,32	2,69
Reims	243	ikea	commerces	0,55	0,00	0,64	0,89	1,78	0,98	1,89
Reims	300	ikea	bar	0,51	0,00	1,26	0,83	1,46	1,03	0,91
Reims	300	ikea	arrêts de bus	0,29	0,00	0,74	1,68	1,23	1,89	1,59
Reims	243	railway	commerces	0,00	0,14	0,54	0,61	1,02	0,72	1,26
Reims	300	railway	bar	0,49	0,00	0,43	0,28	0,72	0,74	0,55
Reims	300	railway	arrêts de bus	0,88	0,00	1,29	0,64	0,85	0,66	1,74
Reims	243	ikea	commerces	0,32	0,39	0,00	1,65	1,12	1,06	1,96
Reims	316	ikea	bar	1,21	0,00	2,09	1,20	1,12	2,12	1,32
Reims	350	ikea	arrêts de bus	0,97	0,00	2,03	1,77	1,82	2,06	1,66
Reims	243	railway	commerces	1,24	0,54	2,26	0,00	1,27	0,00	0,98
Reims	316	railway	bar	0,45	0,00	1,20	0,52	1,46	0,87	1,86
Reims	350	railway	arrêts de bus	0,75	0,00	0,67	0,74	0,94	0,85	1,43
Reims	243	ikea	commerces	0,79	0,00	1,14	0,61	0,71	1,11	1,44
Reims	316	ikea	bar	0,22	0,00	1,22	1,76	1,76	1,46	1,46
Reims	400	ikea	arrêts de bus	0,00	0,02	1,74	0,95	2,14	0,95	0,99
Reims	243	railway	commerces	0,88	0,00	0,51	0,41	0,87	1,22	0,75
Reims	316	railway	bar	2,36	0,00	3,40	1,82	1,57	2,63	3,06
Reims	400	railway	arrêts de bus	0,64	0,00	0,82	0,47	2,06	0,54	1,64
Reims	243	ikea	commerces	0,38	0,00	0,58	1,00	1,12	0,61	0,80
Reims	316	ikea	bar	0,61	0,01	0,36	0,01	0,61	0,00	0,00
Reims	450	ikea	arrêts de bus	0,79	0,00	1,57	0,37	1,60	0,42	1,42
Reims	243	railway	commerces	1,05	0,00	1,07	0,66	0,66	2,70	1,63
Reims	316	railway	bar	0,54	0,00	0,45	0,24	1,03	0,29	0,72
Reims	450	railway	arrêts de bus	0,62	0,00	0,57	1,26	1,35	1,01	1,41
Reims	243	ikea	commerces	0,44	0,00	0,66	1,10	0,68	0,87	0,52
Reims	316	ikea	bar	1,61	0,00	0,89	1,43	1,43	1,42	2,79
Reims	500	ikea	arrêts de bus	1,19	0,00	0,82	1,08	1,08	1,97	1,70
Reims	243	railway	commerces	0,65	0,00	1,65	1,55	1,56	1,59	1,60
Reims	316	railway	bar	0,61	0,18	0,00	0,49	0,42	0,18	0,40

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Reims	500	railway	arrêts de bus	0,84	0,00	1,19	1,48	0,85	0,95	1,48
Rennes	250	ikea	commerces	1,43	0,00	2,68	4,18	0,68	3,00	2,13
Rennes	250	ikea	bar	0,32	0,00	0,37	0,99	0,10	0,35	0,35
Rennes	250	ikea	arrêts de bus	0,72	0,00	0,92	0,34	0,65	0,22	0,83
Rennes	250	railway	commerces	0,59	0,00	0,60	1,80	1,10	0,40	0,74
Rennes	250	railway	bar	0,20	0,00	1,44	0,93	1,78	0,94	1,56
Rennes	250	railway	arrêts de bus	0,71	0,00	0,94	3,20	1,10	1,28	0,78
Rennes	300	ikea	commerces	0,94	0,00	0,92	0,92	1,17	1,41	1,41
Rennes	300	ikea	bar	0,00	0,11	1,08	0,92	0,26	1,11	1,33
Rennes	300	ikea	arrêts de bus	0,61	0,00	0,54	0,81	1,01	1,01	1,01
Rennes	300	railway	commerces	1,42	0,00	0,95	1,15	1,13	1,14	1,58
Rennes	300	railway	bar	1,06	0,00	1,15	1,18	1,78	1,09	0,75
Rennes	300	railway	arrêts de bus	0,95	0,00	1,16	0,19	1,49	1,13	0,81
Rennes	350	ikea	commerces	1,60	0,00	1,69	1,71	1,25	1,22	1,68
Rennes	350	ikea	bar	0,00	0,03	0,59	1,03	1,37	1,05	1,40
Rennes	350	ikea	arrêts de bus	0,29	0,00	1,39	0,65	0,91	1,08	0,70
Rennes	350	railway	commerces	0,17	0,00	1,09	0,24	0,24	0,23	0,48
Rennes	350	railway	bar	1,28	0,19	0,35	0,00	2,06	1,68	1,16
Rennes	350	railway	arrêts de bus	0,40	0,07	0,00	1,20	1,28	1,06	1,23
Rennes	400	ikea	commerces	0,20	0,01	0,49	0,64	0,50	0,00	0,98
Rennes	399	ikea	bar	0,12	0,00	0,88	0,99	0,89	0,23	1,43
Rennes	400	ikea	arrêts de bus	0,57	0,00	0,54	0,63	1,26	0,65	1,03
Rennes	400	railway	commerces	0,71	0,00	1,55	0,23	0,23	1,27	0,71
Rennes	400	railway	bar	0,43	0,07	1,25	0,00	0,20	0,34	1,22
Rennes	400	railway	arrêts de bus	1,05	0,00	1,32	1,45	2,25	1,22	1,22
Rennes	450	ikea	commerces	1,29	0,00	1,68	0,55	1,88	0,55	1,67
Rennes	450	ikea	bar	1,84	0,00	1,16	1,64	1,54	1,34	1,75
Rennes	450	ikea	arrêts de bus	0,71	0,00	1,25	0,77	0,39	0,82	0,82
Rennes	450	railway	commerces	0,40	0,00	0,28	0,91	1,33	1,25	0,98
Rennes	450	railway	bar	0,76	0,00	0,18	0,75	0,51	1,34	2,23
Rennes	450	railway	arrêts de bus	0,79	0,00	1,60	1,53	1,71	1,08	1,56
Rennes	500	ikea	commerces	1,10	0,11	1,35	0,00	0,94	0,00	1,40
Rennes	499	ikea	bar	0,81	0,00	0,65	1,04	0,88	0,88	1,00
Rennes	500	ikea	arrêts de bus	0,75	0,00	1,59	0,89	1,20	0,78	1,82
Rennes	500	railway	commerces	1,13	0,00	1,18	0,69	0,69	0,98	0,67
Rennes	500	railway	bar	0,79	0,00	0,61	0,78	0,78	1,89	2,76
Rennes	500	railway	arrêts de bus	0,00	0,00	0,98	0,83	0,83	0,83	0,83
Rouen	250	ikea	commerces	0,76	0,00	0,84	0,57	0,57	0,94	1,99
Rouen	250	ikea	bar	0,91	0,00	0,62	1,07	1,56	0,77	1,26
Rouen	250	ikea	arrêts de bus	0,86	0,00	1,04	2,63	2,85	0,93	1,04
Rouen	250	railway	commerces	0,65	0,00	0,28	1,76	0,97	0,67	0,67
Rouen	250	railway	bar	1,20	0,26	1,40	0,00	0,12	0,09	1,38
Rouen	250	railway	arrêts de bus	0,59	0,00	0,67	0,71	0,82	1,54	1,11
Rouen	300	ikea	commerces	1,30	0,00	0,96	0,70	0,95	4,14	2,05
Rouen	300	ikea	bar	0,40	0,06	0,00	0,63	0,23	1,16	0,78
Rouen	300	ikea	arrêts de bus	1,49	1,08	2,15	2,65	0,00	3,92	1,68
Rouen	300	railway	commerces	0,21	0,04	0,46	1,37	0,19	0,00	0,00
Rouen	300	railway	bar	0,67	0,00	0,63	3,99	0,65	0,36	0,62
Rouen	300	railway	arrêts de bus	0,01	0,00	0,58	0,68	0,83	0,94	0,67
Rouen	350	ikea	commerces	1,04	0,00	0,84	1,54	1,29	1,45	1,40
Rouen	350	ikea	bar	0,74	0,00	1,05	1,14	1,23	0,40	0,95
Rouen	350	ikea	arrêts de bus	0,28	0,00	0,20	0,53	0,90	0,71	1,64
Rouen	350	railway	commerces	0,16	0,00	1,34	1,79	1,69	0,83	1,39
Rouen	350	railway	bar	2,23	0,00	3,01	7,29	2,67	2,39	2,50
Rouen	349	railway	arrêts de bus	0,63	0,00	0,92	0,22	0,76	0,22	0,55
Rouen	400	ikea	commerces	0,48	0,11	0,00	0,66	0,66	0,51	0,53
Rouen	400	ikea	bar	0,28	0,03	0,44	0,00	0,89	0,76	0,93
Rouen	399	ikea	arrêts de bus	1,09	0,00	1,11	1,46	0,18	0,93	1,92
Rouen	400	railway	commerces	0,42	0,00	1,15	2,12	2,12	1,20	1,41
Rouen	400	railway	bar	0,22	0,00	0,86	1,37	0,70	0,47	0,58
Rouen	399	railway	arrêts de bus	0,80	0,00	0,91	0,95	0,59	0,80	0,45
Rouen	450	ikea	commerces	1,03	0,00	1,68	1,83	1,42	0,99	1,02
Rouen	450	ikea	bar	0,67	0,00	0,84	1,27	1,73	0,96	1,75
Rouen	449	ikea	arrêts de bus	0,66	0,00	1,22	1,09	0,58	0,64	0,56

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Rouen	450	railway	commerces	0,43	0,00	0,54	0,97	0,83	0,05	0,50
Rouen	450	railway	bar	0,24	0,00	1,46	0,66	2,03	0,48	0,60
Rouen	450	railway	arrêts de bus	0,89	0,00	0,95	0,92	1,60	0,79	1,52
Rouen	467	ikea	commerces	0,36	0,00	0,49	1,89	0,68	3,76	1,76
Rouen	500	ikea	bar	0,12	0,00	0,58	0,22	0,95	1,34	0,43
Rouen	498	ikea	arrêts de bus	0,89	0,00	1,73	1,36	1,26	1,41	1,04
Rouen	467	railway	commerces	0,52	0,00	0,69	2,42	2,07	0,88	0,62
Rouen	500	railway	bar	0,85	0,00	0,81	0,60	0,84	0,41	1,37
Rouen	499	railway	arrêts de bus	0,34	0,00	1,29	0,65	0,85	1,09	0,44
Saint-Etienne	250	ikea	commerces	0,03	0,00	0,39	0,10	0,79	0,59	0,16
Saint-Etienne	249	ikea	bar	2,04	1,49	0,00	1,94	2,88	1,86	1,86
Saint-Etienne	250	ikea	arrêts de bus	0,09	0,00	0,77	2,08	0,84	0,58	0,45
Saint-Etienne	250	railway	commerces	0,10	0,00	0,93	0,45	1,12	0,35	0,26
Saint-Etienne	249	railway	bar	1,92	0,00	1,44	1,65	2,72	2,30	2,93
Saint-Etienne	250	railway	arrêts de bus	1,02	0,05	0,70	0,26	0,26	0,42	0,00
Saint-Etienne	300	ikea	commerces	1,25	0,00	1,14	1,48	1,06	0,82	1,29
Saint-Etienne	299	ikea	bar	0,76	0,00	2,21	0,28	1,75	0,28	1,80
Saint-Etienne	300	ikea	arrêts de bus	0,47	0,00	1,67	1,47	1,47	2,38	2,58
Saint-Etienne	300	railway	commerces	1,02	0,98	1,36	1,14	0,00	1,33	1,05
Saint-Etienne	299	railway	bar	0,29	0,00	0,65	0,39	0,39	0,54	0,78
Saint-Etienne	300	railway	arrêts de bus	0,59	0,00	0,88	0,30	0,30	0,36	0,36
Saint-Etienne	300	ikea	commerces	0,51	0,00	1,31	1,15	0,64	1,03	1,09
Saint-Etienne	349	ikea	bar	0,17	0,00	0,69	0,61	0,49	0,47	1,24
Saint-Etienne	349	ikea	arrêts de bus	0,70	0,23	1,28	0,86	0,93	0,00	0,00
Saint-Etienne	300	railway	commerces	0,33	0,25	0,96	0,00	0,86	0,53	0,53
Saint-Etienne	349	railway	bar	0,48	0,00	0,98	0,36	1,50	0,54	0,32
Saint-Etienne	350	railway	arrêts de bus	1,01	0,00	1,03	1,00	1,02	0,92	2,14
Saint-Etienne	300	ikea	commerces	0,00	0,02	0,08	0,85	0,89	0,46	0,23
Saint-Etienne	399	ikea	bar	0,26	0,00	0,99	1,29	1,91	0,56	1,51
Saint-Etienne	400	ikea	arrêts de bus	0,75	0,00	0,63	1,10	1,41	0,46	1,48
Saint-Etienne	300	railway	commerces	0,94	0,00	0,99	0,53	0,38	0,94	0,64
Saint-Etienne	399	railway	bar	0,28	0,08	1,08	0,59	1,01	0,00	1,01
Saint-Etienne	399	railway	arrêts de bus	0,00	0,00	1,18	0,89	1,49	0,52	1,31
Saint-Etienne	300	ikea	commerces	0,12	0,00	0,11	0,28	0,63	0,34	0,33
Saint-Etienne	434	ikea	bar	0,59	0,00	0,98	0,31	0,28	0,31	1,16
Saint-Etienne	449	ikea	arrêts de bus	0,52	0,00	0,49	0,69	0,69	0,65	1,34
Saint-Etienne	300	railway	commerces	0,67	0,00	0,55	0,24	0,57	0,58	0,81
Saint-Etienne	434	railway	bar	0,31	0,00	1,32	0,65	1,26	0,42	0,31
Saint-Etienne	450	railway	arrêts de bus	0,68	0,00	0,44	0,40	0,22	0,38	0,26
Saint-Etienne	300	ikea	commerces	0,07	0,00	0,58	0,33	0,15	0,14	0,13
Saint-Etienne	434	ikea	bar	0,73	0,00	1,38	0,52	0,90	0,69	0,66
Saint-Etienne	500	ikea	arrêts de bus	0,70	0,00	0,37	0,60	1,52	1,07	0,96
Saint-Etienne	300	railway	commerces	0,34	0,52	0,20	0,66	0,30	0,00	1,05
Saint-Etienne	434	railway	bar	1,50	0,10	0,00	1,65	2,06	0,00	1,26
Saint-Etienne	499	railway	arrêts de bus	0,61	0,00	1,03	0,77	0,90	0,72	0,72
Strasbourg	250	ikea	commerces	0,49	0,03	0,56	0,84	0,73	0,68	0,00
Strasbourg	250	ikea	bar	0,25	0,19	0,49	0,26	1,91	0,00	1,51
Strasbourg	250	ikea	arrêts de bus	0,56	0,00	0,69	1,65	0,49	0,75	0,51
Strasbourg	250	railway	commerces	1,11	0,56	0,06	1,89	2,77	0,00	1,53
Strasbourg	250	railway	bar	1,04	1,30	0,99	0,00	0,81	0,38	1,35
Strasbourg	250	railway	arrêts de bus	0,42	0,00	1,30	1,78	1,58	1,16	1,72
Strasbourg	300	ikea	commerces	1,60	0,00	1,62	1,25	2,33	0,33	1,25
Strasbourg	300	ikea	bar	1,36	0,00	1,25	1,15	0,42	1,71	1,71
Strasbourg	300	ikea	arrêts de bus	1,00	0,00	0,36	1,03	1,03	0,22	1,02
Strasbourg	300	railway	commerces	0,09	0,00	0,99	1,62	2,94	1,90	2,04
Strasbourg	300	railway	bar	0,15	0,00	1,33	0,90	1,47	0,51	0,47
Strasbourg	300	railway	arrêts de bus	0,83	0,00	1,49	1,90	2,02	1,22	0,52
Strasbourg	348	ikea	commerces	1,14	0,00	1,64	1,00	1,26	1,46	0,90
Strasbourg	350	ikea	bar	0,16	0,04	1,16	0,00	1,28	0,00	0,50
Strasbourg	350	ikea	arrêts de bus	0,54	0,05	1,91	2,40	0,00	0,75	1,59
Strasbourg	348	railway	commerces	0,27	0,00	1,93	0,76	0,76	1,15	1,15
Strasbourg	350	railway	bar	0,17	0,00	1,07	0,58	0,60	0,36	1,44
Strasbourg	350	railway	arrêts de bus	0,73	0,00	1,19	3,19	3,40	0,95	0,92
Strasbourg	348	ikea	commerces	0,75	0,00	1,62	1,40	1,25	0,34	0,34

Tableau A3 (suite) - Résultats du FG-MNS sur les instances à flotte homogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Strasbourg	400	ikea	bar	1,28	0,00	0,78	1,61	0,82	0,34	1,90
Strasbourg	400	ikea	arrêts de bus	0,80	0,00	0,39	0,60	0,60	0,52	0,74
Strasbourg	348	railway	commerces	2,90	0,00	4,09	4,33	3,59	2,79	3,07
Strasbourg	400	railway	bar	1,09	0,00	1,29	0,44	1,06	0,46	1,61
Strasbourg	400	railway	arrêts de bus	2,05	0,00	1,90	2,35	1,89	1,76	1,76
Strasbourg	348	ikea	commerces	1,16	0,00	0,75	2,20	1,25	0,98	1,08
Strasbourg	450	ikea	bar	1,25	0,00	1,86	1,20	1,85	0,40	1,03
Strasbourg	450	ikea	arrêts de bus	0,15	0,00	0,52	0,72	1,49	0,10	0,41
Strasbourg	348	railway	commerces	1,27	0,00	0,89	1,27	1,95	0,67	1,51
Strasbourg	450	railway	bar	0,73	0,00	1,27	1,14	1,15	0,52	1,26
Strasbourg	450	railway	arrêts de bus	1,12	0,00	1,22	1,00	0,67	0,58	0,29
Strasbourg	348	ikea	commerces	0,32	0,00	0,93	0,74	1,37	0,85	0,81
Strasbourg	500	ikea	bar	0,72	0,00	1,03	0,88	1,37	0,54	0,63
Strasbourg	500	ikea	arrêts de bus	1,62	0,00	1,36	2,01	2,06	1,02	1,55
Strasbourg	348	railway	commerces	0,76	0,00	1,40	2,95	0,94	0,68	1,66
Strasbourg	500	railway	bar	1,08	0,00	1,22	1,43	1,98	0,73	1,24
Strasbourg	500	railway	arrêts de bus	0,34	0,19	0,51	0,40	1,04	0,00	0,00
Toulon	250	ikea	commerces	0,23	0,07	0,87	0,00	0,26	0,88	1,11
Toulon	250	ikea	bar	0,00	0,08	0,33	0,28	0,28	0,25	0,43
Toulon	250	ikea	arrêts de bus	0,29	0,00	0,72	0,42	0,61	0,30	0,23
Toulon	250	railway	commerces	0,36	0,00	0,49	0,27	0,52	0,29	0,67
Toulon	250	railway	bar	0,49	0,00	0,46	0,55	0,40	0,25	0,92
Toulon	250	railway	arrêts de bus	0,55	0,00	0,42	0,55	0,98	0,59	0,80
Toulon	300	ikea	commerces	0,30	0,00	0,70	0,24	0,24	0,36	0,36
Toulon	300	ikea	bar	0,29	0,00	0,91	0,46	0,38	0,17	0,38
Toulon	300	ikea	arrêts de bus	0,00	0,08	0,56	0,36	0,56	0,36	0,56
Toulon	300	railway	commerces	0,92	0,00	1,13	0,41	1,06	2,56	0,33
Toulon	300	railway	bar	0,80	0,00	1,35	0,41	0,83	0,41	0,45
Toulon	300	railway	arrêts de bus	0,81	0,00	0,88	0,74	0,74	0,79	1,01
Toulon	350	ikea	commerces	0,32	0,07	0,50	0,00	0,47	0,47	0,45
Toulon	350	ikea	bar	0,02	0,00	0,94	0,46	0,47	0,49	0,44
Toulon	350	ikea	arrêts de bus	0,64	0,15	0,73	0,52	0,28	0,52	0,00
Toulon	350	railway	commerces	0,48	0,00	0,81	0,60	0,41	0,60	0,72
Toulon	350	railway	bar	0,93	0,33	1,39	0,00	1,52	0,00	0,95
Toulon	350	railway	arrêts de bus	0,28	0,00	0,57	0,47	0,53	0,33	0,42
Toulon	356	ikea	commerces	0,37	0,00	0,41	0,30	0,43	0,33	0,09
Toulon	400	ikea	bar	0,48	0,33	0,31	0,00	0,00	0,96	0,30
Toulon	400	ikea	arrêts de bus	0,61	0,00	0,57	0,43	0,78	0,24	0,78
Toulon	356	railway	commerces	0,08	0,06	0,86	0,00	0,94	0,78	1,46
Toulon	400	railway	bar	0,00	0,87	1,21	1,05	1,05	1,05	1,10
Toulon	400	railway	arrêts de bus	0,54	0,00	0,80	0,47	0,94	0,32	0,83
Toulon	356	ikea	commerces	0,80	0,00	0,55	0,34	0,92	0,24	0,27
Toulon	450	ikea	bar	0,16	0,03	0,87	0,00	0,47	0,19	0,19
Toulon	450	ikea	arrêts de bus	0,76	0,00	0,74	0,91	0,91	0,54	0,39
Toulon	356	railway	commerces	0,59	0,00	1,38	0,66	0,66	0,35	0,54
Toulon	450	railway	bar	0,28	0,00	2,05	0,08	0,78	0,90	1,27
Toulon	450	railway	arrêts de bus	0,42	0,00	0,40	0,31	0,31	0,27	0,23
Toulon	356	ikea	commerces	0,25	0,00	0,01	0,31	0,50	0,53	0,55
Toulon	500	ikea	bar	0,47	0,00	1,95	0,72	1,03	1,48	1,33
Toulon	500	ikea	arrêts de bus	0,72	0,00	0,41	0,56	0,56	0,34	1,02
Toulon	356	railway	commerces	0,43	0,00	0,77	0,22	0,73	0,28	0,59
Toulon	500	railway	bar	0,00	0,16	0,86	0,20	0,70	0,45	0,73
Toulon	500	railway	arrêts de bus	0,38	0,00	0,52	0,59	0,52	0,59	0,35

A.5.2 Flotte hétérogène

Tableau A4 – Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Avignon	193	ikea	commerces	175,60	172,73	175,60	177,27	177,63	175,73	175,73
Avignon	250	ikea	bar	225,03	220,53	225,03	223,93	226,67	224,24	228,11
Avignon	250	ikea	arrêts de bus	215,72	212,33	215,81	214,96	215,99	215,77	218,46
Avignon	193	railway	commerces	159,68	157,43	161,49	160,86	161,89	160,97	160,43
Avignon	250	railway	bar	217,09	213,59	217,16	218,68	218,17	219,07	218,64
Avignon	250	railway	arrêts de bus	216,61	211,36	216,13	215,49	217,01	215,49	216,62
Avignon	193	ikea	commerces	172,29	169,59	172,32	171,23	172,72	171,71	172,54
Avignon	300	ikea	bar	253,45	249,30	253,45	255,08	257,01	251,62	254,09
Avignon	299	ikea	arrêts de bus	240,88	238,72	245,41	245,37	243,42	246,13	249,81
Avignon	193	railway	commerces	192,66	189,69	192,64	192,42	193,15	192,14	192,14
Avignon	300	railway	bar	268,49	264,50	270,42	271,48	270,05	271,10	272,84
Avignon	300	railway	arrêts de bus	263,46	259,43	266,01	263,26	267,57	265,00	262,28
Avignon	193	ikea	commerces	164,99	162,07	166,27	164,76	164,28	164,76	164,76
Avignon	350	ikea	bar	302,64	296,12	303,82	300,89	305,71	302,68	302,82
Avignon	350	ikea	arrêts de bus	287,92	282,41	290,63	290,13	289,31	291,63	290,10
Avignon	193	railway	commerces	170,99	169,03	170,99	174,01	171,89	172,87	173,03
Avignon	350	railway	bar	311,48	305,76	312,09	311,50	311,52	314,65	314,24
Avignon	350	railway	arrêts de bus	286,63	281,41	286,63	284,50	286,80	286,80	286,80
Avignon	193	ikea	commerces	182,74	178,01	184,24	184,05	182,74	183,75	186,06
Avignon	400	ikea	bar	351,70	348,21	349,09	355,27	356,77	357,46	353,24
Avignon	400	ikea	arrêts de bus	330,13	326,16	332,34	333,36	336,96	335,28	330,17
Avignon	193	railway	commerces	176,13	171,58	174,85	174,76	174,76	174,76	174,76
Avignon	400	railway	bar	362,63	354,73	362,63	367,89	366,60	360,18	369,39
Avignon	399	railway	arrêts de bus	360,01	355,12	363,66	365,90	365,90	373,90	362,97
Avignon	193	ikea	commerces	184,18	181,37	184,18	183,39	185,63	185,95	184,06
Avignon	450	ikea	bar	374,64	367,06	374,98	374,74	372,87	372,92	379,24
Avignon	449	ikea	arrêts de bus	363,10	358,89	363,10	366,62	364,85	370,90	368,52
Avignon	193	railway	commerces	202,20	200,05	202,20	204,07	203,55	204,36	202,33
Avignon	450	railway	bar	399,98	388,87	393,41	399,55	396,97	401,64	400,19
Avignon	449	railway	arrêts de bus	356,86	349,71	363,91	363,57	357,50	361,97	361,97
Avignon	193	ikea	commerces	185,29	182,98	185,37	188,87	186,17	187,79	187,22
Avignon	464	ikea	bar	369,41	359,61	369,41	371,86	367,89	373,75	371,18
Avignon	499	ikea	arrêts de bus	386,26	379,63	393,34	393,77	394,70	391,88	391,88
Avignon	193	railway	commerces	180,97	177,60	181,31	180,40	180,03	180,40	180,40
Avignon	464	railway	bar	407,78	402,62	408,74	410,27	413,02	414,59	414,01
Avignon	499	railway	arrêts de bus	405,97	399,00	405,97	410,56	407,46	410,56	410,56
Bordeaux	250	ikea	commerces	303,93	294,12	303,79	298,05	304,80	304,87	307,83
Bordeaux	248	ikea	bar	245,70	241,05	248,57	246,78	247,81	246,85	249,79
Bordeaux	247	ikea	arrêts de bus	320,15	314,72	325,87	324,48	324,48	322,26	326,91
Bordeaux	250	railway	commerces	317,34	309,68	319,87	317,04	311,94	311,65	319,29
Bordeaux	248	railway	bar	220,03	214,41	222,11	220,35	219,81	219,33	219,33
Bordeaux	249	railway	arrêts de bus	306,43	302,47	310,36	305,51	309,18	307,52	307,52
Bordeaux	300	ikea	commerces	363,75	355,89	367,71	372,46	363,61	366,09	367,86
Bordeaux	297	ikea	bar	285,24	276,56	285,60	285,69	282,79	282,79	282,79
Bordeaux	296	ikea	arrêts de bus	383,38	375,29	394,15	389,95	383,98	387,52	388,62
Bordeaux	300	railway	commerces	344,31	336,07	338,01	345,07	341,84	344,95	348,07
Bordeaux	297	railway	bar	275,48	268,47	275,48	270,94	272,71	275,01	279,51
Bordeaux	297	railway	arrêts de bus	422,45	414,08	426,34	424,88	432,00	427,36	429,30
Bordeaux	350	ikea	commerces	420,48	411,89	425,32	423,43	423,43	423,43	423,43
Bordeaux	346	ikea	bar	318,48	306,69	318,48	321,57	318,06	319,96	318,52
Bordeaux	346	ikea	arrêts de bus	429,39	421,30	430,33	430,98	432,10	431,76	428,77
Bordeaux	350	railway	commerces	379,20	370,89	379,20	378,49	381,67	378,11	378,11
Bordeaux	347	railway	bar	306,80	298,37	306,25	305,60	305,60	300,59	305,60
Bordeaux	348	railway	arrêts de bus	417,87	402,01	418,35	409,44	414,30	414,63	417,84
Bordeaux	400	ikea	commerces	472,68	465,89	483,29	482,00	485,10	487,80	484,99
Bordeaux	398	ikea	bar	372,69	364,15	378,60	378,17	378,15	377,31	375,75
Bordeaux	398	ikea	arrêts de bus	525,55	505,50	526,81	519,24	514,66	519,24	519,24
Bordeaux	400	railway	commerces	407,46	400,82	419,20	409,33	410,09	409,43	409,43
Bordeaux	398	railway	bar	347,77	345,06	353,56	353,21	344,03	352,80	352,38
Bordeaux	399	railway	arrêts de bus	477,62	466,86	482,58	483,22	486,13	486,14	484,43

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Bordeaux	450	ikea	commerces	527,37	507,73	534,19	527,75	527,75	524,42	523,41
Bordeaux	444	ikea	bar	446,63	433,33	450,28	444,86	444,86	444,86	443,58
Bordeaux	443	ikea	arrêts de bus	533,00	522,79	535,88	541,77	542,58	542,58	542,58
Bordeaux	450	railway	commerces	459,94	452,35	459,94	464,08	464,08	461,20	464,08
Bordeaux	446	railway	bar	427,44	414,96	428,33	432,73	421,68	428,90	438,03
Bordeaux	448	railway	arrêts de bus	533,50	520,68	534,87	534,66	523,94	527,88	534,66
Bordeaux	500	ikea	commerces	565,13	550,98	569,19	567,99	570,62	567,90	567,90
Bordeaux	497	ikea	bar	454,61	445,33	456,13	455,23	455,23	455,23	455,23
Bordeaux	496	ikea	arrêts de bus	615,13	599,47	622,82	625,14	616,96	627,11	631,39
Bordeaux	500	railway	commerces	480,08	465,27	480,08	480,90	481,51	483,38	479,61
Bordeaux	494	railway	bar	457,35	446,83	460,69	461,24	460,60	465,00	457,02
Bordeaux	497	railway	arrêts de bus	635,68	611,71	636,95	633,24	641,41	641,58	646,18
Brest	249	ikea	arrêts de bus	325,12	319,80	325,12	327,82	327,01	325,40	328,82
Brest	249	railway	arrêts de bus	445,70	428,71	446,08	439,37	439,37	440,37	451,37
Brest	300	ikea	arrêts de bus	401,81	394,47	401,81	407,98	412,79	404,72	412,88
Brest	349	ikea	arrêts de bus	407,95	400,27	406,78	409,14	406,46	409,14	410,46
Brest	350	railway	arrêts de bus	614,27	599,00	614,27	614,49	613,36	615,33	615,33
Brest	450	ikea	arrêts de bus	534,08	529,83	534,08	541,61	539,92	539,72	540,35
Brest	499	ikea	arrêts de bus	605,52	593,10	605,84	605,33	605,33	605,33	603,45
ClermontFerrand	250	ikea	commerces	250,67	241,98	250,76	249,52	249,65	249,52	250,62
ClermontFerrand	249	ikea	bar	231,77	228,11	235,86	233,53	236,64	235,43	232,48
ClermontFerrand	250	ikea	arrêts de bus	236,59	232,13	236,59	237,77	237,77	238,58	237,83
ClermontFerrand	250	railway	commerces	242,74	236,52	242,74	242,81	243,22	241,05	246,22
ClermontFerrand	250	railway	bar	224,80	220,63	227,82	227,12	224,00	225,36	225,95
ClermontFerrand	250	railway	arrêts de bus	245,88	240,53	243,74	242,37	248,51	245,63	246,30
ClermontFerrand	297	ikea	commerces	288,89	285,17	288,89	290,79	293,45	291,83	292,78
ClermontFerrand	299	ikea	bar	282,51	276,03	285,51	282,65	282,23	280,78	284,77
ClermontFerrand	300	ikea	arrêts de bus	302,71	303,02	307,56	306,17	306,17	309,87	309,87
ClermontFerrand	297	railway	commerces	273,54	266,04	274,05	270,67	272,93	271,32	276,97
ClermontFerrand	300	railway	bar	239,03	233,68	241,75	239,61	239,52	240,38	241,99
ClermontFerrand	300	railway	arrêts de bus	276,21	268,29	275,69	274,33	273,31	274,09	272,08
ClermontFerrand	297	ikea	commerces	295,76	290,39	295,76	290,43	297,01	297,37	298,40
ClermontFerrand	350	ikea	bar	316,40	309,10	317,54	321,63	318,51	319,05	316,98
ClermontFerrand	350	ikea	arrêts de bus	334,48	330,19	339,20	332,63	332,63	335,13	337,55
ClermontFerrand	297	railway	commerces	264,94	260,43	267,03	267,20	268,91	268,70	268,91
ClermontFerrand	349	railway	bar	292,13	285,34	293,95	292,49	292,94	293,52	298,92
ClermontFerrand	350	railway	arrêts de bus	313,95	311,80	316,42	317,54	315,83	321,62	318,83
ClermontFerrand	297	ikea	commerces	277,54	272,37	277,94	277,18	277,18	279,50	280,01
ClermontFerrand	399	ikea	bar	352,63	340,81	354,61	352,83	349,42	354,84	350,89
ClermontFerrand	400	ikea	arrêts de bus	386,03	378,39	386,03	386,15	386,43	387,46	387,32
ClermontFerrand	297	railway	commerces	270,25	265,01	270,25	273,24	274,31	273,07	273,92
ClermontFerrand	399	railway	bar	324,04	313,12	324,04	323,76	322,72	326,38	324,71
ClermontFerrand	400	railway	arrêts de bus	335,67	330,54	334,16	338,34	338,34	337,12	337,33
ClermontFerrand	297	ikea	commerces	290,77	287,93	295,30	296,89	291,50	293,37	294,96
ClermontFerrand	449	ikea	bar	398,32	386,40	397,86	396,05	397,25	403,22	402,81
ClermontFerrand	450	ikea	arrêts de bus	404,68	397,41	408,18	397,55	397,55	398,36	409,20
ClermontFerrand	297	railway	commerces	259,26	251,72	258,63	259,01	261,36	259,74	261,56
ClermontFerrand	450	railway	bar	361,52	353,47	361,52	364,75	364,75	365,96	363,69
ClermontFerrand	450	railway	arrêts de bus	360,02	354,47	362,59	362,21	364,14	365,55	365,72
ClermontFerrand	297	ikea	commerces	292,70	288,13	292,70	290,89	296,44	293,51	293,27
ClermontFerrand	499	ikea	bar	432,25	418,31	432,25	433,49	433,49	431,71	431,71
ClermontFerrand	500	ikea	arrêts de bus	476,56	466,80	478,27	478,86	476,94	486,04	481,81
ClermontFerrand	297	railway	commerces	259,15	252,20	260,21	259,28	259,28	256,89	259,28
ClermontFerrand	499	railway	bar	385,58	375,67	387,74	385,91	388,38	390,15	383,58
ClermontFerrand	500	railway	arrêts de bus	386,35	377,23	384,85	383,93	383,93	384,82	384,82
Dijon	250	ikea	commerces	312,69	306,04	312,69	312,55	310,47	312,79	314,51
Dijon	249	ikea	bar	254,00	250,23	257,07	253,43	253,43	259,05	257,22
Dijon	250	ikea	arrêts de bus	222,07	217,07	222,72	218,09	222,61	221,26	224,33
Dijon	250	railway	commerces	301,47	300,20	301,47	306,67	310,36	309,37	308,54
Dijon	249	railway	bar	198,90	194,70	201,23	202,06	199,88	199,27	201,44
Dijon	250	railway	arrêts de bus	172,67	170,69	174,75	171,60	175,25	172,81	174,14
Dijon	271	ikea	commerces	334,78	327,09	336,55	334,98	330,54	337,07	332,93
Dijon	299	ikea	bar	305,91	299,94	307,29	302,55	306,12	309,30	304,33
Dijon	300	ikea	arrêts de bus	250,76	243,19	250,76	248,58	250,26	249,92	251,42

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Dijon	271	railway	commerces	325,48	319,76	325,79	323,73	322,99	323,13	323,73
Dijon	299	railway	bar	251,76	244,54	252,63	248,23	250,31	251,03	251,04
Dijon	300	railway	arrêts de bus	227,10	226,09	227,10	231,31	231,31	226,71	230,62
Dijon	271	ikea	commerces	329,08	323,64	332,33	333,46	333,18	332,49	334,24
Dijon	349	ikea	bar	330,59	323,83	332,86	328,15	331,68	330,29	333,65
Dijon	349	ikea	arrêts de bus	268,28	262,92	269,15	269,60	269,89	273,14	272,55
Dijon	271	railway	commerces	344,27	341,60	347,36	348,40	348,90	346,90	348,90
Dijon	349	railway	bar	286,99	280,56	289,45	287,25	287,71	286,78	287,54
Dijon	350	railway	arrêts de bus	249,81	245,71	252,78	249,88	249,88	251,85	254,04
Dijon	271	ikea	commerces	321,19	315,07	321,19	319,46	319,60	319,67	320,66
Dijon	400	ikea	bar	347,89	340,67	348,86	352,86	354,50	348,06	360,78
Dijon	400	ikea	arrêts de bus	296,66	292,77	296,66	296,63	303,22	297,34	297,34
Dijon	271	railway	commerces	313,24	307,68	313,24	311,65	312,18	312,66	309,73
Dijon	400	railway	bar	321,40	310,26	321,40	315,52	319,31	319,67	321,80
Dijon	399	railway	arrêts de bus	263,33	260,34	263,33	262,41	264,19	264,59	264,63
Dijon	271	ikea	commerces	317,10	313,78	319,79	314,91	320,42	313,25	321,73
Dijon	449	ikea	bar	411,38	402,46	416,02	407,96	413,88	411,16	423,50
Dijon	449	ikea	arrêts de bus	346,14	340,42	349,50	348,74	348,76	345,75	344,42
Dijon	271	railway	commerces	303,58	299,69	305,78	303,98	301,32	303,21	308,03
Dijon	449	railway	bar	355,29	344,68	360,35	355,75	354,62	357,39	357,22
Dijon	449	railway	arrêts de bus	302,87	299,08	303,12	304,85	304,19	303,45	305,52
Dijon	271	ikea	commerces	343,51	337,81	343,56	342,85	345,57	345,57	345,57
Dijon	470	ikea	bar	398,70	391,81	400,77	402,37	405,34	404,27	402,58
Dijon	499	ikea	arrêts de bus	379,85	370,05	379,85	378,66	380,93	385,42	380,41
Dijon	271	railway	commerces	326,61	321,45	326,85	325,77	325,82	325,44	326,77
Dijon	470	railway	bar	363,30	357,44	373,16	367,49	372,37	366,95	370,84
Dijon	499	railway	arrêts de bus	352,55	348,05	354,47	356,18	356,18	354,42	361,67
Grenoble	250	ikea	commerces	544,29	527,66	554,52	538,49	549,79	552,24	554,16
Grenoble	250	ikea	bar	519,99	512,14	523,87	520,93	524,52	512,75	526,01
Grenoble	250	ikea	arrêts de bus	644,40	624,24	646,29	648,11	651,68	642,00	638,85
Grenoble	250	railway	commerces	1 193,82	1 167,02	1 193,82	1 198,83	1 209,81	1 217,56	1 204,70
Grenoble	250	railway	bar	1 084,04	1 073,41	1 077,90	1 079,45	1 088,51	1 081,54	1 093,42
Grenoble	249	railway	arrêts de bus	1 318,38	1 287,93	1 318,38	1 342,16	1 335,16	1 320,52	1 320,52
Grenoble	300	ikea	commerces	637,63	622,72	639,80	641,59	642,05	644,26	644,33
Grenoble	300	ikea	bar	655,98	644,35	657,69	658,96	655,38	656,44	660,64
Grenoble	300	ikea	arrêts de bus	769,28	755,47	772,09	767,96	767,75	771,80	772,10
Grenoble	300	railway	commerces	1 386,47	1 374,09	1 399,82	1 402,36	1 418,72	1 404,42	1 417,13
Grenoble	300	railway	bar	1 399,58	1 377,50	1 410,36	1 397,33	1 388,47	1 423,89	1 402,30
Grenoble	300	railway	arrêts de bus	1 604,66	1 567,64	1 604,66	1 606,47	1 604,66	1 607,75	1 604,27
Grenoble	350	ikea	commerces	702,94	694,22	709,38	711,53	723,54	723,51	723,21
Grenoble	349	ikea	bar	705,23	684,89	704,87	701,45	701,45	702,51	704,96
Grenoble	350	ikea	arrêts de bus	820,03	795,63	821,44	816,19	816,19	814,69	809,91
Grenoble	350	railway	commerces	1 607,21	1 587,00	1 610,30	1 603,78	1 613,05	1 622,72	1 616,43
Grenoble	350	railway	bar	1 588,14	1 564,64	1 592,54	1 588,84	1 592,54	1 588,22	1 597,64
Grenoble	350	railway	arrêts de bus	1 837,94	1 820,78	1 848,10	1 847,51	1 847,93	1 831,30	1 831,30
Grenoble	400	ikea	commerces	800,00	774,27	796,46	794,84	810,57	793,54	801,53
Grenoble	399	ikea	bar	824,30	817,99	833,45	837,00	840,21	840,95	837,55
Grenoble	400	ikea	arrêts de bus	926,56	908,03	926,56	936,69	926,67	923,81	935,17
Grenoble	400	railway	commerces	1 944,20	1 917,15	1 977,87	1 974,18	1 977,27	1 949,48	1 958,74
Grenoble	400	railway	bar	1 741,13	1 693,18	1 749,36	1 732,93	1 763,11	1 763,24	1 748,03
Grenoble	400	railway	arrêts de bus	1 928,36	1 878,69	1 912,54	1 904,33	1 904,33	1 904,33	1 904,33
Grenoble	450	ikea	commerces	901,78	872,27	898,01	892,66	889,67	892,66	892,66
Grenoble	450	ikea	bar	912,68	894,55	927,84	920,70	921,00	911,47	917,91
Grenoble	450	ikea	arrêts de bus	1 098,65	1 077,52	1 115,16	1 102,81	1 102,81	1 109,02	1 109,02
Grenoble	450	railway	commerces	2 065,57	2 025,13	2 065,57	2 064,68	2 066,62	2 066,62	2 066,62
Grenoble	450	railway	bar	1 788,13	1 765,78	1 788,13	1 789,06	1 796,21	1 810,26	1 822,28
Grenoble	450	railway	arrêts de bus	2 049,71	2 006,68	2 059,06	2 050,33	2 065,55	2 101,23	2 103,58
Grenoble	500	ikea	commerces	973,45	945,72	973,45	988,07	988,59	972,77	975,31
Grenoble	500	ikea	bar	981,57	960,65	988,10	980,10	991,01	991,04	991,04
Grenoble	499	ikea	arrêts de bus	1 107,77	1 062,40	1 104,43	1 092,24	1 095,33	1 090,62	1 107,24
Grenoble	500	railway	commerces	2 108,03	2 057,62	2 100,73	2 051,69	2 095,06	2 057,20	2 112,54
Grenoble	500	railway	bar	2 183,69	2 152,95	2 188,08	2 188,28	2 214,05	2 195,79	2 197,17
Grenoble	500	railway	arrêts de bus	2 378,51	2 353,12	2 395,02	2 396,04	2 416,11	2 410,96	2 410,96
Lyon	250	ikea	commerces	325,53	320,23	326,40	328,24	329,93	328,24	329,75

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Lyon	247	ikea	bar	301,72	294,17	301,72	298,93	299,32	298,17	301,37
Lyon	244	ikea	arrêts de bus	334,62	323,54	338,78	333,38	333,38	329,74	329,74
Lyon	250	railway	commerces	338,70	333,17	342,64	341,54	341,74	342,74	341,80
Lyon	247	railway	bar	351,95	345,29	351,95	355,65	355,65	354,21	353,44
Lyon	249	railway	arrêts de bus	404,84	397,22	408,36	406,80	407,30	409,59	411,55
Lyon	300	ikea	commerces	370,35	362,27	374,70	369,21	369,21	370,89	369,23
Lyon	298	ikea	bar	353,93	346,96	356,35	355,77	354,53	356,06	357,26
Lyon	294	ikea	arrêts de bus	435,75	426,22	446,76	437,75	440,94	441,78	443,02
Lyon	300	railway	commerces	449,92	438,62	449,92	451,24	451,32	453,63	455,80
Lyon	297	railway	bar	382,48	374,73	384,50	382,63	382,63	386,37	377,28
Lyon	296	railway	arrêts de bus	455,85	448,06	457,34	451,16	453,10	459,01	458,82
Lyon	350	ikea	commerces	423,23	416,14	431,31	424,63	429,90	426,93	426,00
Lyon	347	ikea	bar	374,56	366,43	374,76	375,13	373,62	375,50	373,21
Lyon	346	ikea	arrêts de bus	461,72	453,40	467,59	462,70	465,54	465,54	463,47
Lyon	350	railway	commerces	515,53	506,35	515,53	514,76	519,95	521,34	521,80
Lyon	347	railway	bar	430,82	423,14	430,82	437,14	437,14	437,27	437,28
Lyon	344	railway	arrêts de bus	527,18	527,34	542,22	541,37	532,84	540,92	540,92
Lyon	400	ikea	commerces	490,61	481,46	490,61	494,72	494,72	493,69	493,69
Lyon	396	ikea	bar	418,11	408,19	419,96	413,72	424,08	420,86	424,60
Lyon	397	ikea	arrêts de bus	525,11	510,18	525,77	526,34	527,08	521,72	526,32
Lyon	400	railway	commerces	559,51	548,39	559,51	553,28	555,39	555,35	555,35
Lyon	396	railway	bar	461,07	448,83	461,57	462,86	462,86	462,86	462,86
Lyon	394	railway	arrêts de bus	579,44	570,54	572,92	574,84	585,53	583,77	594,10
Lyon	450	ikea	commerces	581,25	565,62	588,33	575,43	587,61	575,04	581,39
Lyon	445	ikea	bar	434,06	424,68	434,06	435,31	435,31	436,67	440,48
Lyon	440	ikea	arrêts de bus	562,22	548,25	567,76	565,08	574,15	570,61	574,15
Lyon	450	railway	commerces	632,92	619,93	632,92	634,91	632,36	634,60	635,08
Lyon	445	railway	bar	547,57	534,05	547,57	540,29	540,29	550,99	546,16
Lyon	439	railway	arrêts de bus	645,69	633,36	645,44	646,78	641,91	646,78	646,78
Lyon	500	ikea	commerces	583,00	569,01	584,44	578,87	578,87	577,50	579,81
Lyon	494	ikea	bar	537,67	530,13	537,67	538,11	542,11	554,46	543,18
Lyon	492	ikea	arrêts de bus	612,37	599,07	612,37	621,60	615,01	615,01	621,26
Lyon	500	railway	commerces	648,62	638,03	648,60	649,15	649,30	647,34	649,62
Lyon	494	railway	bar	576,39	570,22	579,28	579,13	600,16	594,76	585,90
Lyon	494	railway	arrêts de bus	728,12	709,03	722,29	725,30	725,51	724,95	728,89
Marseille	250	ikea	commerces	275,24	268,10	275,89	270,28	270,28	271,98	270,28
Marseille	250	ikea	bar	281,18	277,47	285,22	284,19	282,72	284,30	286,32
Marseille	250	ikea	arrêts de bus	300,96	292,99	301,06	299,81	301,15	302,35	304,10
Marseille	250	railway	commerces	455,14	438,20	458,82	454,89	455,87	459,38	459,73
Marseille	250	railway	bar	403,77	395,14	407,38	404,94	402,68	406,75	406,26
Marseille	250	railway	arrêts de bus	433,38	419,48	433,35	432,97	435,85	432,64	431,94
Marseille	300	ikea	commerces	319,67	313,41	322,47	318,57	323,43	320,71	323,48
Marseille	300	ikea	bar	319,61	315,29	319,61	320,79	317,86	325,98	320,28
Marseille	300	ikea	arrêts de bus	350,02	344,41	352,49	352,52	353,50	350,17	355,94
Marseille	300	railway	commerces	449,52	442,71	452,13	454,20	458,22	454,49	458,09
Marseille	300	railway	bar	504,63	488,84	506,23	491,26	497,95	495,96	497,26
Marseille	300	railway	arrêts de bus	513,63	497,07	509,58	512,57	511,78	513,93	511,27
Marseille	350	ikea	commerces	348,78	346,39	348,78	354,13	354,13	356,10	356,10
Marseille	350	ikea	bar	358,44	349,71	355,69	360,38	361,00	357,19	362,90
Marseille	350	ikea	arrêts de bus	400,51	394,66	403,19	403,30	401,92	402,30	402,10
Marseille	350	railway	commerces	529,94	522,62	529,94	543,56	543,56	543,56	533,67
Marseille	350	railway	bar	550,73	535,66	550,73	557,94	557,94	555,65	548,38
Marseille	350	railway	arrêts de bus	568,84	552,92	572,02	566,33	565,48	569,64	566,41
Marseille	400	ikea	commerces	395,99	393,67	406,01	399,82	396,53	410,19	402,90
Marseille	400	ikea	bar	396,92	383,30	395,71	393,45	396,65	396,77	400,97
Marseille	400	ikea	arrêts de bus	442,12	431,95	444,85	443,20	444,05	445,96	442,46
Marseille	400	railway	commerces	613,94	602,10	624,10	615,10	615,10	626,91	610,52
Marseille	400	railway	bar	613,62	598,17	616,08	615,83	622,35	612,87	636,36
Marseille	400	railway	arrêts de bus	645,47	630,59	645,47	646,33	650,72	649,09	648,58
Marseille	450	ikea	commerces	424,31	419,50	424,31	428,45	428,71	428,71	428,71
Marseille	450	ikea	bar	453,38	442,81	453,38	455,27	458,98	459,51	459,34
Marseille	450	ikea	arrêts de bus	479,17	469,86	484,22	485,57	483,81	483,66	486,79
Marseille	450	railway	commerces	670,04	657,44	662,59	672,30	672,30	665,82	693,29
Marseille	450	railway	bar	689,01	673,60	695,93	686,49	702,78	686,46	723,02

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Marseille	450	railway	arrêts de bus	708,49	696,56	708,49	717,84	708,76	715,42	718,75
Marseille	500	ikea	commerces	510,79	495,53	510,79	511,62	507,16	513,22	507,38
Marseille	500	ikea	bar	476,22	463,82	476,22	476,36	476,36	472,05	472,05
Marseille	500	ikea	arrêts de bus	520,92	510,42	524,03	524,99	525,81	525,09	525,81
Marseille	500	railway	commerces	775,57	759,25	776,44	777,97	770,97	777,97	774,46
Marseille	500	railway	bar	801,49	789,36	801,49	824,29	819,13	824,29	824,29
Marseille	500	railway	arrêts de bus	779,53	751,71	776,70	782,48	777,24	776,95	775,51
Metz	218	ikea	commerces	235,86	230,28	239,57	235,98	239,22	237,75	236,09
Metz	249	ikea	bar	233,91	228,02	235,42	234,85	235,77	235,56	234,87
Metz	250	ikea	arrêts de bus	266,00	258,35	266,00	266,70	266,58	266,71	265,34
Metz	218	railway	commerces	246,17	239,80	247,35	248,60	247,72	247,72	248,45
Metz	248	railway	bar	230,14	221,10	227,96	230,92	228,52	229,82	230,52
Metz	250	railway	arrêts de bus	247,02	241,97	247,02	250,03	248,73	251,45	251,04
Metz	218	ikea	commerces	256,89	252,31	258,88	257,02	258,27	260,45	258,27
Metz	298	ikea	bar	255,33	249,88	255,65	256,92	257,08	255,96	259,29
Metz	300	ikea	arrêts de bus	290,43	282,82	292,63	290,01	290,37	292,52	291,91
Metz	218	railway	commerces	260,30	253,17	259,67	259,75	257,56	258,89	257,74
Metz	299	railway	bar	268,34	261,38	270,00	268,01	269,55	269,93	269,87
Metz	300	railway	arrêts de bus	293,48	288,43	295,27	296,97	297,75	296,57	296,50
Metz	218	ikea	commerces	228,21	222,68	229,85	226,85	229,32	230,13	230,38
Metz	348	ikea	bar	312,16	308,86	315,61	312,33	315,25	312,39	312,10
Metz	350	ikea	arrêts de bus	354,42	343,94	358,35	353,07	351,98	353,07	353,07
Metz	218	railway	commerces	259,49	254,21	258,07	259,60	260,57	261,79	261,28
Metz	348	railway	bar	338,99	332,48	341,26	338,37	342,71	344,79	340,41
Metz	350	railway	arrêts de bus	332,69	328,71	332,69	335,16	340,17	339,60	337,17
Metz	218	ikea	commerces	227,35	223,56	231,08	228,16	228,69	228,86	234,11
Metz	398	ikea	bar	343,84	336,28	345,76	342,86	343,74	345,13	346,03
Metz	400	ikea	arrêts de bus	402,55	393,98	402,55	408,04	408,75	396,23	403,66
Metz	218	railway	commerces	253,17	244,09	254,86	255,62	252,47	253,28	255,45
Metz	398	railway	bar	388,91	372,64	382,49	385,91	391,93	395,21	390,25
Metz	400	railway	arrêts de bus	349,20	343,46	352,10	349,74	352,14	352,98	353,06
Metz	218	ikea	commerces	245,51	242,35	246,30	246,04	249,64	247,29	249,70
Metz	414	ikea	bar	362,74	355,99	367,74	361,27	361,27	363,35	353,92
Metz	450	ikea	arrêts de bus	454,08	439,60	457,73	452,22	459,48	457,47	455,98
Metz	218	railway	commerces	264,90	262,61	264,17	265,14	261,84	265,07	265,30
Metz	414	railway	bar	378,04	372,47	379,35	386,01	381,23	380,81	383,69
Metz	450	railway	arrêts de bus	424,09	412,90	421,52	420,30	420,92	427,82	426,88
Metz	218	ikea	commerces	246,20	239,80	248,17	248,69	246,48	246,34	248,27
Metz	414	ikea	bar	357,13	346,77	359,93	358,60	358,60	360,52	361,19
Metz	500	ikea	arrêts de bus	458,56	447,70	459,67	457,67	449,90	459,87	459,87
Metz	218	railway	commerces	241,69	236,06	241,99	241,60	239,47	241,90	244,53
Metz	414	railway	bar	379,60	371,08	380,35	376,41	383,14	380,99	382,61
Metz	500	railway	arrêts de bus	443,20	434,88	450,83	445,46	445,92	445,46	447,68
Montpellier	250	ikea	commerces	364,49	356,05	364,49	365,66	366,54	368,50	363,92
Montpellier	250	ikea	bar	322,90	316,69	320,92	321,60	322,51	322,51	322,51
Montpellier	250	ikea	arrêts de bus	359,77	349,44	360,27	356,87	360,48	355,80	360,67
Montpellier	250	railway	commerces	481,08	479,01	481,31	482,37	482,37	482,37	482,37
Montpellier	250	railway	bar	437,56	430,86	437,56	441,81	441,01	449,52	443,60
Montpellier	250	railway	arrêts de bus	495,80	485,89	495,80	494,44	496,43	496,43	496,25
Montpellier	300	ikea	commerces	388,19	380,95	388,08	388,16	392,63	387,80	391,28
Montpellier	300	ikea	bar	369,05	364,27	369,05	369,00	372,00	377,67	382,61
Montpellier	300	ikea	arrêts de bus	427,09	419,74	428,48	426,31	426,31	429,75	426,86
Montpellier	300	railway	commerces	578,84	570,65	579,09	583,22	586,27	577,92	577,92
Montpellier	300	railway	bar	466,78	452,84	469,72	475,81	464,82	472,37	473,70
Montpellier	300	railway	arrêts de bus	586,34	575,85	586,04	589,94	596,12	592,08	592,06
Montpellier	350	ikea	commerces	465,57	449,46	469,05	463,52	465,16	463,25	463,25
Montpellier	350	ikea	bar	440,16	432,09	440,16	444,44	444,44	444,31	439,67
Montpellier	350	ikea	arrêts de bus	509,84	489,91	511,21	509,39	509,39	508,17	508,16
Montpellier	350	railway	commerces	617,64	608,00	620,69	623,21	619,48	629,92	631,47
Montpellier	350	railway	bar	565,27	560,93	565,27	563,68	563,68	575,34	568,60
Montpellier	350	railway	arrêts de bus	624,91	619,04	624,91	628,43	628,91	629,00	629,00
Montpellier	400	ikea	commerces	526,60	517,89	526,60	528,13	536,57	539,80	555,31
Montpellier	400	ikea	bar	474,12	463,93	480,45	471,13	472,29	471,22	468,34
Montpellier	400	ikea	arrêts de bus	539,00	528,66	542,83	529,75	535,27	532,79	532,79

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Montpellier	400	railway	commerces	676,02	657,50	681,77	677,41	677,81	683,78	673,28
Montpellier	400	railway	bar	656,15	641,01	665,13	651,91	654,76	668,52	671,92
Montpellier	400	railway	arrêts de bus	692,66	668,53	693,70	685,99	685,99	685,99	679,89
Montpellier	450	ikea	commerces	567,46	560,42	574,93	572,26	572,26	566,60	579,70
Montpellier	450	ikea	bar	548,94	534,96	553,13	550,97	553,20	557,16	565,54
Montpellier	450	ikea	arrêts de bus	600,30	587,13	602,83	601,88	607,45	609,32	607,66
Montpellier	450	railway	commerces	726,78	703,71	735,24	728,95	727,64	735,52	725,78
Montpellier	450	railway	bar	715,95	706,98	717,80	717,69	726,49	719,74	719,74
Montpellier	450	railway	arrêts de bus	784,01	766,50	784,01	785,97	783,73	784,72	783,00
Montpellier	500	ikea	commerces	601,04	580,76	607,69	605,04	601,41	601,08	606,29
Montpellier	500	ikea	bar	626,92	602,63	627,69	623,73	623,73	650,34	628,52
Montpellier	500	ikea	arrêts de bus	607,12	597,03	613,91	608,52	608,52	611,26	611,26
Montpellier	500	railway	commerces	843,39	832,03	843,39	846,90	852,17	851,33	860,72
Montpellier	500	railway	bar	831,98	826,94	831,98	838,17	838,17	852,03	855,57
Montpellier	500	railway	arrêts de bus	834,15	806,21	822,97	822,97	822,97	822,97	822,97
Nantes	250	ikea	commerces	374,92	361,10	374,32	368,23	368,23	373,73	373,73
Nantes	249	ikea	bar	318,07	308,97	314,92	315,29	315,29	316,54	321,86
Nantes	249	ikea	arrêts de bus	357,73	354,47	362,38	359,51	361,81	361,81	361,81
Nantes	250	railway	commerces	536,99	523,58	536,99	535,51	538,04	535,66	540,15
Nantes	249	railway	bar	460,36	453,17	472,48	456,78	466,43	468,45	468,71
Nantes	250	railway	arrêts de bus	543,40	538,51	553,39	546,00	550,29	546,68	546,68
Nantes	300	ikea	commerces	413,28	405,15	413,28	410,96	409,20	413,96	411,09
Nantes	299	ikea	bar	365,01	353,19	368,52	367,02	367,02	368,24	369,96
Nantes	300	ikea	arrêts de bus	363,58	358,75	366,22	369,67	367,03	371,44	369,76
Nantes	300	railway	commerces	653,80	641,44	653,72	654,18	659,84	659,84	656,52
Nantes	299	railway	bar	584,32	571,81	578,16	581,03	581,03	581,03	581,17
Nantes	300	railway	arrêts de bus	667,82	651,40	664,01	665,91	668,67	669,23	664,71
Nantes	350	ikea	commerces	450,79	435,88	450,79	445,73	451,37	452,72	458,57
Nantes	349	ikea	bar	395,62	390,46	394,49	393,30	400,64	393,30	403,57
Nantes	350	ikea	arrêts de bus	480,94	469,00	480,49	476,51	476,51	478,92	478,92
Nantes	350	railway	commerces	726,71	706,03	719,71	727,31	726,98	725,24	727,61
Nantes	349	railway	bar	703,93	679,32	703,93	697,75	697,75	697,75	703,37
Nantes	349	railway	arrêts de bus	759,60	747,87	764,06	754,03	758,68	758,68	758,68
Nantes	400	ikea	commerces	502,40	491,62	508,44	500,73	503,48	500,73	510,98
Nantes	399	ikea	bar	432,80	420,58	438,03	433,31	434,28	439,43	432,35
Nantes	400	ikea	arrêts de bus	517,49	503,76	523,81	523,39	509,31	523,39	523,39
Nantes	400	railway	commerces	864,76	851,29	864,76	874,55	875,36	884,65	880,58
Nantes	399	railway	bar	783,55	771,48	785,05	784,01	792,07	805,28	791,31
Nantes	400	railway	arrêts de bus	855,60	843,78	855,60	855,69	855,69	855,69	855,69
Nantes	450	ikea	commerces	554,22	536,30	548,31	547,01	550,30	547,01	553,73
Nantes	449	ikea	bar	490,89	479,08	491,33	487,24	493,39	487,99	487,37
Nantes	450	ikea	arrêts de bus	547,23	536,28	552,92	550,89	559,30	548,97	548,43
Nantes	450	railway	commerces	942,52	908,34	935,40	935,68	944,08	945,74	939,77
Nantes	449	railway	bar	829,15	809,31	829,20	826,86	821,30	826,86	826,86
Nantes	450	railway	arrêts de bus	923,07	903,40	925,62	920,46	920,46	920,46	920,46
Nantes	500	ikea	commerces	579,83	565,34	587,09	584,76	584,76	594,00	590,84
Nantes	499	ikea	bar	555,51	541,96	558,43	547,31	556,43	549,26	554,35
Nantes	499	ikea	arrêts de bus	624,97	609,13	623,55	622,52	625,82	622,52	622,52
Nantes	500	railway	commerces	1 004,07	992,25	1 004,07	1 014,76	1 014,12	1 003,43	1 014,76
Nantes	499	railway	bar	994,32	977,43	1 009,72	993,33	1 003,10	998,95	999,09
Nantes	500	railway	arrêts de bus	1 028,58	1 022,56	1 062,09	1 059,71	1 062,28	1 040,83	1 054,44
Orleans	250	ikea	commerces	386,36	380,40	386,36	387,62	391,28	386,58	381,82
Orleans	250	ikea	bar	381,18	370,01	387,14	378,30	378,30	384,46	384,46
Orleans	250	ikea	arrêts de bus	307,96	305,71	312,50	312,34	318,81	312,63	320,85
Orleans	250	railway	commerces	613,11	605,56	605,15	613,86	618,77	619,64	621,02
Orleans	249	railway	bar	535,83	521,32	533,93	539,99	538,67	546,71	544,43
Orleans	250	railway	arrêts de bus	456,83	454,49	456,83	468,18	465,93	463,88	465,07
Orleans	300	ikea	commerces	468,91	466,72	481,85	480,98	480,98	479,45	477,99
Orleans	300	ikea	bar	408,19	401,19	414,59	405,74	410,60	408,14	408,14
Orleans	300	ikea	arrêts de bus	362,18	357,98	365,45	356,44	365,10	363,16	365,10
Orleans	300	railway	commerces	691,22	668,12	686,46	677,03	686,76	677,03	687,85
Orleans	299	railway	bar	607,30	593,53	609,47	601,25	607,29	610,06	608,10
Orleans	300	railway	arrêts de bus	540,70	519,51	540,70	544,28	529,94	538,13	538,13
Orleans	330	ikea	commerces	503,92	496,39	504,05	510,97	507,02	509,17	504,55

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Orleans	350	ikea	bar	490,09	475,76	495,98	509,62	488,69	490,66	506,57
Orleans	350	ikea	arrêts de bus	419,09	406,36	412,69	417,89	417,89	419,85	408,82
Orleans	330	railway	commerces	704,87	692,31	704,21	700,23	701,47	705,96	709,97
Orleans	350	railway	bar	676,28	662,77	676,28	679,64	680,95	681,98	681,98
Orleans	350	railway	arrêts de bus	666,86	645,28	663,68	657,26	655,62	663,21	679,67
Orleans	330	ikea	commerces	506,74	498,36	506,74	509,66	512,01	508,74	511,05
Orleans	399	ikea	bar	551,20	531,68	555,16	552,80	542,13	548,62	562,01
Orleans	400	ikea	arrêts de bus	468,01	456,38	468,66	470,99	475,33	476,58	466,79
Orleans	330	railway	commerces	742,31	719,48	732,91	737,32	738,48	738,48	737,16
Orleans	399	railway	bar	754,11	750,90	754,11	751,70	751,70	751,70	751,70
Orleans	400	railway	arrêts de bus	743,23	723,07	743,23	751,46	727,47	748,43	763,13
Orleans	330	ikea	commerces	495,48	490,85	497,05	492,58	495,95	491,96	491,96
Orleans	449	ikea	bar	571,43	561,36	571,43	572,23	573,34	584,95	584,95
Orleans	450	ikea	arrêts de bus	506,56	496,56	506,56	508,22	508,22	509,27	505,04
Orleans	330	railway	commerces	719,85	710,95	732,40	720,61	731,72	724,15	720,00
Orleans	449	railway	bar	913,37	897,45	913,37	933,54	939,43	930,42	920,97
Orleans	450	railway	arrêts de bus	892,32	860,40	897,28	886,55	892,97	891,19	899,29
Orleans	330	ikea	commerces	497,60	493,92	499,76	498,61	492,03	496,28	496,04
Orleans	499	ikea	bar	649,45	634,12	663,22	658,13	652,75	657,59	647,71
Orleans	500	ikea	arrêts de bus	578,13	564,83	578,13	581,38	575,86	582,91	577,21
Orleans	330	railway	commerces	672,14	668,11	676,15	670,67	674,58	674,24	674,58
Orleans	499	railway	bar	960,71	927,46	963,30	959,39	951,75	959,61	950,24
Orleans	500	railway	arrêts de bus	911,09	878,29	908,85	893,00	900,18	910,83	911,01
Reims	243	ikea	commerces	271,70	266,30	272,10	272,89	278,33	271,95	273,88
Reims	250	ikea	bar	278,80	272,82	279,42	278,13	277,68	279,95	280,52
Reims	250	ikea	arrêts de bus	204,33	197,67	202,83	204,18	204,18	203,08	203,12
Reims	243	railway	commerces	422,67	407,95	418,98	416,54	410,89	410,89	416,08
Reims	250	railway	bar	399,59	389,68	394,88	402,13	399,81	395,80	395,80
Reims	250	railway	arrêts de bus	387,05	380,21	393,71	387,52	385,79	381,23	383,98
Reims	243	ikea	commerces	265,32	255,20	265,73	261,67	263,75	266,04	266,29
Reims	300	ikea	bar	304,00	292,37	300,88	296,76	301,60	298,79	299,43
Reims	300	ikea	arrêts de bus	253,16	246,77	250,38	253,77	250,51	255,58	256,29
Reims	243	railway	commerces	378,01	370,62	383,65	374,72	384,10	374,72	384,27
Reims	300	railway	bar	497,74	485,82	497,74	498,91	501,75	498,30	499,04
Reims	300	railway	arrêts de bus	436,27	428,80	436,27	442,95	442,95	436,19	440,17
Reims	243	ikea	commerces	262,27	259,07	261,65	261,53	262,01	261,51	262,01
Reims	316	ikea	bar	304,91	293,74	307,67	298,90	298,90	298,90	298,90
Reims	350	ikea	arrêts de bus	264,18	258,06	265,70	266,22	263,49	264,68	267,36
Reims	243	railway	commerces	401,11	396,18	401,11	402,06	403,37	403,12	398,35
Reims	316	railway	bar	489,45	479,16	487,68	495,95	495,92	491,23	493,26
Reims	350	railway	arrêts de bus	486,59	479,52	486,59	494,18	491,60	488,66	491,92
Reims	243	ikea	commerces	260,90	259,42	260,90	266,81	265,38	264,58	263,96
Reims	316	ikea	bar	301,28	295,88	301,28	304,01	303,54	305,95	303,43
Reims	400	ikea	arrêts de bus	311,56	304,71	311,56	311,91	310,99	314,00	314,00
Reims	243	railway	commerces	399,72	395,18	401,37	399,96	399,96	402,40	402,40
Reims	316	railway	bar	532,04	529,98	546,66	547,57	544,56	549,31	537,65
Reims	400	railway	arrêts de bus	608,85	593,60	610,10	617,24	605,12	615,71	618,19
Reims	243	ikea	commerces	268,19	265,06	269,61	269,03	267,76	267,72	268,05
Reims	316	ikea	bar	315,83	306,19	317,11	311,31	313,15	314,52	311,52
Reims	450	ikea	arrêts de bus	337,61	328,24	338,91	338,59	337,63	339,95	339,39
Reims	243	railway	commerces	402,08	398,49	404,01	403,43	403,66	402,20	402,49
Reims	316	railway	bar	516,62	506,45	516,62	513,89	517,14	520,67	519,39
Reims	450	railway	arrêts de bus	658,86	628,05	664,08	653,58	653,58	651,61	659,27
Reims	243	ikea	commerces	266,44	262,91	266,44	266,76	266,76	266,76	265,77
Reims	316	ikea	bar	308,94	304,42	309,31	312,26	306,84	311,26	312,26
Reims	500	ikea	arrêts de bus	369,03	359,61	375,63	368,05	368,89	369,61	372,54
Reims	243	railway	commerces	377,33	371,55	380,76	379,41	377,70	382,71	381,61
Reims	316	railway	bar	562,71	550,32	560,29	569,33	566,46	564,86	573,62
Reims	500	railway	arrêts de bus	702,25	682,56	700,54	705,69	709,32	707,27	701,36
Rennes	250	ikea	commerces	428,71	418,43	428,71	426,67	430,50	426,11	428,21
Rennes	250	ikea	bar	349,53	343,90	357,28	355,61	355,61	354,46	351,19
Rennes	250	ikea	arrêts de bus	391,01	387,41	391,01	391,62	396,25	391,50	399,97
Rennes	250	railway	commerces	357,32	351,80	357,86	360,93	360,93	356,75	358,08
Rennes	250	railway	bar	379,90	373,93	379,90	379,59	381,46	382,32	380,35

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Rennes	250	railway	arrêts de bus	370,84	366,05	374,16	369,67	375,81	373,99	382,78
Rennes	300	ikea	commerces	448,64	436,32	449,25	440,36	446,41	444,91	445,15
Rennes	300	ikea	bar	423,07	413,78	425,18	421,61	422,26	418,41	422,05
Rennes	300	ikea	arrêts de bus	470,92	461,59	470,92	473,94	472,92	470,16	472,32
Rennes	300	railway	commerces	439,78	431,87	439,78	443,52	445,28	449,95	449,95
Rennes	299	railway	bar	406,83	392,15	406,83	407,97	403,87	404,36	413,25
Rennes	300	railway	arrêts de bus	435,53	425,87	439,28	436,40	439,15	434,28	434,28
Rennes	350	ikea	commerces	508,64	496,03	514,14	513,83	510,45	515,68	514,20
Rennes	349	ikea	bar	497,21	488,00	499,41	497,68	497,68	497,68	497,68
Rennes	350	ikea	arrêts de bus	542,23	534,39	551,64	547,98	542,46	552,13	552,33
Rennes	350	railway	commerces	471,55	458,14	471,55	468,18	472,45	481,78	467,77
Rennes	349	railway	bar	478,11	461,35	478,99	478,52	478,52	478,52	478,52
Rennes	350	railway	arrêts de bus	491,93	479,39	493,17	490,76	486,32	496,35	491,19
Rennes	400	ikea	commerces	601,97	593,47	604,72	604,87	604,87	604,87	604,87
Rennes	400	ikea	bar	531,22	525,49	522,12	533,52	533,52	533,52	533,52
Rennes	400	ikea	arrêts de bus	614,24	600,36	614,24	613,83	617,10	616,82	618,36
Rennes	400	railway	commerces	544,11	530,45	544,11	546,91	546,91	544,93	532,60
Rennes	400	railway	bar	517,54	500,08	529,84	520,16	536,70	524,21	513,30
Rennes	400	railway	arrêts de bus	545,62	538,07	553,11	542,70	553,72	550,22	558,74
Rennes	450	ikea	commerces	633,11	626,03	633,11	643,62	638,47	637,06	647,54
Rennes	449	ikea	bar	597,73	583,60	598,87	593,44	593,44	593,44	593,44
Rennes	450	ikea	arrêts de bus	633,49	619,16	634,44	632,28	634,46	634,46	634,46
Rennes	450	railway	commerces	584,64	570,72	594,32	582,75	580,25	581,96	590,36
Rennes	449	railway	bar	562,49	537,84	562,29	560,37	558,98	554,31	560,37
Rennes	450	railway	arrêts de bus	577,13	561,11	575,84	577,65	588,26	580,79	580,79
Rennes	500	ikea	commerces	686,67	668,11	686,67	678,17	678,17	689,48	691,52
Rennes	499	ikea	bar	627,12	612,54	624,14	632,59	628,32	630,44	623,26
Rennes	500	ikea	arrêts de bus	696,95	688,88	696,95	701,09	701,61	704,87	700,59
Rennes	500	railway	commerces	657,84	639,89	670,55	653,11	653,11	659,76	667,88
Rennes	498	railway	bar	614,90	598,21	620,54	624,13	613,22	623,02	623,02
Rennes	500	railway	arrêts de bus	643,94	633,73	643,94	645,18	645,18	651,84	650,13
Rouen	250	ikea	commerces	443,80	431,25	442,97	431,93	431,93	439,67	442,50
Rouen	250	ikea	bar	445,65	437,75	446,52	448,10	452,31	441,35	455,52
Rouen	250	ikea	arrêts de bus	413,20	409,22	413,20	416,81	416,81	414,30	422,82
Rouen	250	railway	commerces	294,53	295,14	294,53	297,60	298,01	300,46	302,65
Rouen	250	railway	bar	293,00	282,21	294,75	286,85	291,48	286,04	284,57
Rouen	250	railway	arrêts de bus	305,85	305,50	315,50	306,90	306,90	314,72	309,01
Rouen	300	ikea	commerces	515,57	503,60	529,85	524,55	524,08	519,11	519,35
Rouen	300	ikea	bar	506,86	486,07	506,86	502,00	502,00	499,85	495,53
Rouen	299	ikea	arrêts de bus	504,81	492,14	504,81	508,80	506,41	512,37	507,96
Rouen	300	railway	commerces	356,63	351,09	356,11	356,32	357,03	360,67	358,82
Rouen	300	railway	bar	323,21	316,31	323,21	324,77	324,37	325,07	325,46
Rouen	300	railway	arrêts de bus	397,10	391,99	400,31	397,07	397,07	397,19	401,48
Rouen	350	ikea	commerces	542,70	528,72	542,70	543,42	550,03	549,01	550,03
Rouen	350	ikea	bar	595,19	569,22	591,62	595,45	592,30	595,45	588,34
Rouen	350	ikea	arrêts de bus	580,65	572,12	588,15	585,04	587,24	583,87	583,84
Rouen	350	railway	commerces	389,96	380,62	389,96	393,03	390,90	392,11	386,04
Rouen	350	railway	bar	345,50	332,02	342,44	346,03	346,26	346,10	349,01
Rouen	349	railway	arrêts de bus	435,62	425,95	437,09	436,35	436,85	441,08	441,96
Rouen	400	ikea	commerces	728,81	713,90	747,84	723,02	745,11	748,88	725,05
Rouen	400	ikea	bar	659,98	647,45	665,60	667,80	677,51	677,53	683,76
Rouen	399	ikea	arrêts de bus	657,31	650,66	657,61	660,64	662,17	664,01	664,01
Rouen	400	railway	commerces	449,53	434,57	450,36	451,80	451,80	450,00	448,05
Rouen	400	railway	bar	393,46	382,35	394,19	392,82	392,24	392,60	394,00
Rouen	400	railway	arrêts de bus	475,23	464,08	475,15	469,14	477,91	479,92	480,10
Rouen	450	ikea	commerces	737,50	723,74	743,59	734,00	734,58	745,01	750,35
Rouen	450	ikea	bar	722,93	716,17	742,32	725,76	739,26	750,84	750,84
Rouen	450	ikea	arrêts de bus	707,81	702,25	707,81	723,56	718,14	723,56	723,56
Rouen	450	railway	commerces	511,90	498,15	508,20	511,89	511,89	506,00	509,90
Rouen	450	railway	bar	404,09	394,49	406,50	400,38	404,38	402,07	405,23
Rouen	449	railway	arrêts de bus	564,77	550,30	561,24	562,31	570,18	573,23	573,23
Rouen	467	ikea	commerces	781,26	761,93	780,26	783,88	787,33	785,24	782,57
Rouen	500	ikea	bar	828,89	810,55	841,50	822,42	822,42	821,38	829,60
Rouen	499	ikea	arrêts de bus	804,00	780,10	807,81	807,08	807,08	816,15	794,27

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Rouen	467	railway	commerces	520,01	510,28	519,51	519,21	523,53	518,99	518,99
Rouen	500	railway	bar	482,23	471,61	487,92	483,24	484,06	486,84	488,28
Rouen	500	railway	arrêts de bus	551,29	540,96	554,27	555,22	554,23	557,34	547,07
SaintEtienne	250	ikea	commerces	263,22	258,06	264,51	265,35	267,41	265,61	269,27
SaintEtienne	249	ikea	bar	274,31	264,21	275,03	268,75	267,64	272,15	277,31
SaintEtienne	250	ikea	arrêts de bus	256,54	252,53	256,05	257,75	259,05	256,08	261,65
SaintEtienne	250	railway	commerces	275,72	270,23	275,72	273,86	277,04	275,56	275,56
SaintEtienne	250	railway	bar	303,57	295,15	306,05	296,91	303,00	302,56	302,52
SaintEtienne	250	railway	arrêts de bus	303,59	297,55	302,45	302,96	302,49	303,42	302,66
SaintEtienne	300	ikea	commerces	309,73	302,06	311,03	306,84	311,08	303,76	310,00
SaintEtienne	299	ikea	bar	302,29	295,76	303,82	303,01	303,15	302,78	302,94
SaintEtienne	300	ikea	arrêts de bus	326,45	320,35	331,53	324,85	324,85	328,61	329,15
SaintEtienne	300	railway	commerces	348,09	346,41	360,68	356,83	356,91	357,60	361,39
SaintEtienne	299	railway	bar	365,03	357,51	366,32	363,75	363,75	370,41	369,16
SaintEtienne	300	railway	arrêts de bus	358,52	353,24	360,38	359,31	358,05	359,71	360,03
SaintEtienne	300	ikea	commerces	297,92	292,47	300,62	301,07	302,63	299,23	300,94
SaintEtienne	349	ikea	bar	355,82	346,48	355,82	354,23	355,84	355,84	355,84
SaintEtienne	350	ikea	arrêts de bus	354,23	347,68	352,85	357,08	355,66	357,83	357,83
SaintEtienne	300	railway	commerces	357,18	350,63	359,14	358,10	359,05	358,68	359,05
SaintEtienne	349	railway	bar	401,48	395,17	401,48	405,16	404,83	406,69	400,51
SaintEtienne	350	railway	arrêts de bus	391,61	385,34	391,61	390,54	391,86	390,54	390,30
SaintEtienne	300	ikea	commerces	306,76	299,02	308,05	304,77	304,80	306,28	306,28
SaintEtienne	399	ikea	bar	406,68	391,33	406,68	404,01	403,26	402,35	405,32
SaintEtienne	400	ikea	arrêts de bus	380,63	376,22	386,84	384,44	384,44	385,92	385,92
SaintEtienne	300	railway	commerces	346,50	336,65	346,50	350,26	348,16	346,24	347,10
SaintEtienne	399	railway	bar	469,48	459,43	469,66	462,55	467,54	462,55	464,90
SaintEtienne	400	railway	arrêts de bus	455,24	444,80	456,75	452,72	459,63	458,63	459,63
SaintEtienne	300	ikea	commerces	303,02	298,92	305,92	305,94	302,83	300,45	305,34
SaintEtienne	434	ikea	bar	422,07	410,46	426,32	420,33	422,71	424,95	419,15
SaintEtienne	449	ikea	arrêts de bus	443,67	434,42	448,66	439,31	445,31	444,77	445,62
SaintEtienne	300	railway	commerces	347,16	343,67	351,76	347,32	348,36	347,32	348,99
SaintEtienne	434	railway	bar	484,89	469,18	485,88	483,94	483,94	484,56	484,56
SaintEtienne	450	railway	arrêts de bus	475,23	460,92	473,35	473,23	476,50	471,69	471,69
SaintEtienne	300	ikea	commerces	308,48	302,42	308,48	310,45	308,84	310,35	312,62
SaintEtienne	434	ikea	bar	430,80	425,34	435,63	436,45	436,66	432,32	429,56
SaintEtienne	500	ikea	arrêts de bus	480,90	463,21	482,93	473,75	473,75	478,57	481,44
SaintEtienne	300	railway	commerces	349,60	346,39	351,80	352,94	351,15	352,94	352,94
SaintEtienne	434	railway	bar	482,97	476,98	482,97	490,06	489,63	488,14	497,13
SaintEtienne	500	railway	arrêts de bus	540,04	529,63	542,30	538,76	546,73	548,51	540,18
Strasbourg	250	ikea	commerces	172,73	168,90	174,43	171,38	168,17	173,24	173,72
Strasbourg	250	ikea	bar	159,65	155,92	158,53	159,86	161,69	160,84	159,60
Strasbourg	250	ikea	arrêts de bus	209,86	205,81	209,01	210,39	210,71	207,90	207,90
Strasbourg	250	railway	commerces	195,26	190,79	195,72	196,88	193,98	199,42	197,36
Strasbourg	250	railway	bar	140,69	139,76	142,04	142,07	141,61	142,56	142,73
Strasbourg	250	railway	arrêts de bus	187,34	185,65	187,34	188,87	189,07	191,26	189,00
Strasbourg	300	ikea	commerces	230,36	227,81	230,77	232,06	229,85	231,88	232,25
Strasbourg	300	ikea	bar	182,66	179,10	183,54	185,00	183,26	181,86	186,60
Strasbourg	300	ikea	arrêts de bus	257,18	251,01	259,04	252,74	252,74	257,22	257,39
Strasbourg	300	railway	commerces	216,79	212,22	216,79	217,10	219,73	219,20	218,42
Strasbourg	300	railway	bar	187,28	182,99	189,12	185,20	186,66	185,48	186,79
Strasbourg	300	railway	arrêts de bus	257,63	252,62	261,30	256,86	260,47	256,86	258,73
Strasbourg	348	ikea	commerces	249,88	240,85	249,88	248,43	248,43	250,72	247,72
Strasbourg	350	ikea	bar	220,90	219,33	224,30	226,50	225,09	224,98	224,51
Strasbourg	350	ikea	arrêts de bus	269,68	267,06	269,68	271,63	270,93	272,42	271,55
Strasbourg	348	railway	commerces	260,58	251,43	259,64	260,46	259,45	257,26	260,01
Strasbourg	350	railway	bar	217,61	211,66	217,69	216,68	218,27	215,55	215,55
Strasbourg	350	railway	arrêts de bus	272,47	264,63	272,47	269,91	273,51	270,95	272,21
Strasbourg	348	ikea	commerces	255,52	252,39	257,34	256,34	256,74	257,31	257,34
Strasbourg	400	ikea	bar	253,92	249,27	254,85	251,96	256,69	255,60	256,01
Strasbourg	400	ikea	arrêts de bus	311,45	307,28	312,45	308,70	314,39	313,05	312,78
Strasbourg	348	railway	commerces	278,52	274,79	278,38	283,48	283,47	281,31	282,40
Strasbourg	400	railway	bar	232,93	227,75	234,71	234,26	235,48	234,15	236,06
Strasbourg	400	railway	arrêts de bus	299,71	294,25	299,61	300,27	300,27	300,27	300,27
Strasbourg	348	ikea	commerces	253,22	246,64	250,43	252,54	254,59	250,84	253,61

Tableau A4 (suite) - Résultats du FG-MNS sur les instances à flotte hétérogène

Ville	N	Dépôt	Clients	Écart à la meilleure solution obtenue (%)						
				Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6
Strasbourg	450	ikea	bar	281,93	275,23	281,93	280,13	280,13	280,13	280,13
Strasbourg	450	ikea	arrêts de bus	369,85	362,71	374,11	368,25	376,03	373,76	372,49
Strasbourg	348	railway	commerces	276,45	270,06	276,19	276,51	277,04	277,57	277,57
Strasbourg	450	railway	bar	252,54	248,71	252,54	255,40	254,78	252,56	254,54
Strasbourg	450	railway	arrêts de bus	346,83	340,63	346,01	343,02	343,02	344,12	344,12
Strasbourg	348	ikea	commerces	261,54	257,20	263,95	261,74	266,38	261,74	261,76
Strasbourg	500	ikea	bar	298,54	294,34	298,54	302,41	302,41	304,74	301,69
Strasbourg	500	ikea	arrêts de bus	388,02	376,73	387,53	386,94	385,83	389,56	389,41
Strasbourg	348	railway	commerces	257,41	250,70	257,41	257,48	254,48	258,23	255,65
Strasbourg	500	railway	bar	303,47	298,36	309,03	305,75	304,14	304,82	307,33
Strasbourg	500	railway	arrêts de bus	377,86	369,64	380,92	373,00	374,91	374,66	377,76
Toulon	250	ikea	commerces	387,42	385,81	387,42	388,81	393,33	391,42	394,32
Toulon	250	ikea	bar	362,80	356,96	363,56	363,24	369,20	361,65	372,10
Toulon	250	ikea	arrêts de bus	281,12	277,06	281,67	282,91	282,91	282,87	282,91
Toulon	250	railway	commerces	579,70	568,10	582,47	591,02	581,37	585,86	591,02
Toulon	250	railway	bar	545,74	532,98	546,95	542,27	545,39	548,07	541,52
Toulon	250	railway	arrêts de bus	500,86	491,34	497,94	494,71	500,98	509,17	498,22
Toulon	300	ikea	commerces	389,57	383,44	389,57	390,86	388,63	393,26	397,90
Toulon	300	ikea	bar	405,44	401,91	411,66	407,06	407,06	411,57	412,83
Toulon	300	ikea	arrêts de bus	331,90	325,65	332,50	332,42	331,06	333,21	333,21
Toulon	300	railway	commerces	709,52	699,41	709,52	704,09	713,57	717,00	720,57
Toulon	300	railway	bar	689,11	675,86	683,29	701,37	695,59	696,44	696,44
Toulon	300	railway	arrêts de bus	614,79	604,65	620,97	615,04	614,94	614,87	615,04
Toulon	350	ikea	commerces	469,47	463,77	472,38	471,54	473,11	473,11	468,96
Toulon	350	ikea	bar	496,16	484,38	496,16	500,44	494,03	500,56	497,53
Toulon	350	ikea	arrêts de bus	374,64	366,38	374,70	375,71	375,71	374,22	372,34
Toulon	350	railway	commerces	809,26	794,46	809,26	805,69	805,69	805,68	814,05
Toulon	350	railway	bar	747,09	730,26	740,73	743,30	747,20	746,53	745,67
Toulon	350	railway	arrêts de bus	692,48	686,12	698,79	699,93	694,34	699,93	697,95
Toulon	356	ikea	commerces	491,98	483,68	493,90	492,21	489,82	494,13	488,81
Toulon	400	ikea	bar	558,78	545,97	575,10	560,41	561,37	570,80	562,15
Toulon	400	ikea	arrêts de bus	418,61	411,95	424,23	422,78	422,78	422,78	420,86
Toulon	356	railway	commerces	776,84	767,33	785,50	787,59	787,73	780,59	790,36
Toulon	400	railway	bar	861,28	838,93	878,39	867,13	858,31	869,02	857,79
Toulon	400	railway	arrêts de bus	856,01	838,58	861,72	854,95	860,95	862,34	855,96
Toulon	356	ikea	commerces	489,51	481,43	489,51	489,00	489,81	489,81	488,18
Toulon	450	ikea	bar	579,88	568,38	585,16	582,03	583,17	583,23	581,36
Toulon	450	ikea	arrêts de bus	491,22	485,45	502,54	497,54	502,07	492,83	502,07
Toulon	356	railway	commerces	733,58	717,21	733,58	732,47	727,55	734,62	732,47
Toulon	450	railway	bar	987,42	964,76	987,42	991,74	1 000,96	1 007,16	1 009,63
Toulon	450	railway	arrêts de bus	865,72	852,82	860,57	865,97	872,11	872,70	872,70
Toulon	356	ikea	commerces	499,69	493,26	501,71	496,42	500,29	500,29	500,29
Toulon	500	ikea	bar	637,34	628,12	633,31	637,09	651,12	647,71	646,60
Toulon	500	ikea	arrêts de bus	517,64	506,60	520,06	514,75	514,75	514,75	514,75
Toulon	356	railway	commerces	751,93	747,70	756,41	755,33	761,05	774,33	772,19
Toulon	500	railway	bar	1 090,67	1 086,55	1 119,13	1 104,63	1 108,01	1 127,78	1 126,50
Toulon	500	railway	arrêts de bus	975,04	968,41	975,04	977,79	977,79	977,79	977,79

A.6 Paramétrage de l'arbre

Tableau A5 – Performance des arbres selon les jeux de paramètres

($\bar{F} = \frac{\text{Rappel} + \text{Précision}}{2}$)

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
300	3	FALSE	∞	0,727	0,770	0,567	0,668	0,591	0,488	0,405	0,447
300	3	TRUE	∞	0,717	0,737	0,708	0,723	0,601	0,492	0,417	0,455
300	1	FALSE	∞	0,679	0,714	0,513	0,613	0,576	0,482	0,390	0,436
300	1	TRUE	∞	0,665	0,683	0,659	0,671	0,577	0,502	0,394	0,448
300	7	FALSE	∞	0,786	0,828	0,638	0,733	0,597	0,488	0,412	0,450
300	7	TRUE	∞	0,775	0,808	0,758	0,783	0,593	0,520	0,412	0,466
300	12	FALSE	∞	0,830	0,876	0,694	0,785	0,596	0,461	0,406	0,434
300	12	TRUE	∞	0,837	0,861	0,821	0,841	0,597	0,458	0,408	0,433
300	26	FALSE	∞	0,890	0,921	0,786	0,853	0,610	0,432	0,418	0,425
300	26	TRUE	∞	0,898	0,925	0,877	0,901	0,603	0,443	0,412	0,427
300	38	FALSE	∞	0,908	0,939	0,813	0,876	0,612	0,422	0,419	0,420
300	38	TRUE	∞	0,919	0,944	0,898	0,921	0,615	0,421	0,423	0,422
300	3	FALSE	1	0,551	0,571	0,383	0,477	0,534	0,549	0,367	0,458
300	3	TRUE	1	0,557	0,575	0,554	0,565	0,532	0,555	0,366	0,461
300	1	FALSE	1	0,447	0,843	0,359	0,601	0,446	0,839	0,359	0,599
300	1	TRUE	1	0,547	0,845	0,529	0,687	0,446	0,839	0,359	0,599
300	7	FALSE	1	0,554	0,651	0,397	0,524	0,532	0,612	0,376	0,494
300	7	TRUE	1	0,579	0,653	0,568	0,611	0,530	0,611	0,374	0,493
300	12	FALSE	1	0,518	0,776	0,388	0,582	0,489	0,700	0,362	0,531
300	12	TRUE	1	0,583	0,777	0,559	0,668	0,489	0,700	0,362	0,531
300	26	FALSE	1	0,569	0,774	0,420	0,597	0,519	0,679	0,377	0,528
300	26	TRUE	1	0,621	0,774	0,592	0,683	0,522	0,676	0,378	0,527
300	38	FALSE	1	0,582	0,770	0,429	0,600	0,529	0,672	0,382	0,527
300	38	TRUE	1	0,629	0,775	0,599	0,687	0,526	0,675	0,381	0,528
300	3	FALSE	2	0,589	0,552	0,412	0,482	0,560	0,506	0,380	0,443
300	3	TRUE	2	0,579	0,516	0,590	0,553	0,574	0,477	0,387	0,432
300	1	FALSE	2	0,520	0,669	0,376	0,523	0,542	0,527	0,369	0,448
300	1	TRUE	2	0,558	0,670	0,547	0,609	0,542	0,527	0,369	0,448
300	7	FALSE	2	0,600	0,630	0,431	0,530	0,558	0,548	0,386	0,467
300	7	TRUE	2	0,610	0,675	0,597	0,636	0,545	0,585	0,381	0,483
300	12	FALSE	2	0,601	0,620	0,431	0,526	0,555	0,515	0,378	0,446
300	12	TRUE	2	0,616	0,711	0,597	0,654	0,533	0,585	0,372	0,479
300	26	FALSE	2	0,646	0,719	0,479	0,599	0,563	0,536	0,388	0,462
300	26	TRUE	2	0,670	0,763	0,643	0,703	0,548	0,581	0,383	0,482
300	38	FALSE	2	0,662	0,716	0,495	0,605	0,575	0,529	0,397	0,463
300	38	TRUE	2	0,680	0,749	0,658	0,704	0,562	0,577	0,393	0,485
300	3	FALSE	3	0,576	0,651	0,413	0,532	0,544	0,575	0,379	0,477
300	3	TRUE	3	0,597	0,651	0,587	0,619	0,547	0,576	0,381	0,478
300	1	FALSE	3	0,500	0,778	0,378	0,578	0,493	0,739	0,370	0,554
300	1	TRUE	3	0,570	0,780	0,549	0,665	0,493	0,739	0,370	0,554
300	7	FALSE	3	0,639	0,586	0,467	0,526	0,585	0,494	0,400	0,447
300	7	TRUE	3	0,628	0,678	0,616	0,647	0,560	0,559	0,389	0,474
300	12	FALSE	3	0,638	0,687	0,470	0,579	0,567	0,528	0,390	0,459
300	12	TRUE	3	0,659	0,775	0,629	0,702	0,534	0,617	0,378	0,498
300	26	FALSE	3	0,705	0,748	0,542	0,645	0,567	0,523	0,389	0,456
300	26	TRUE	3	0,724	0,798	0,695	0,746	0,554	0,567	0,386	0,477
300	38	FALSE	3	0,718	0,737	0,558	0,647	0,577	0,474	0,389	0,432
300	38	TRUE	3	0,736	0,823	0,701	0,762	0,557	0,549	0,385	0,467
300	3	FALSE	4	0,636	0,590	0,464	0,527	0,584	0,468	0,396	0,432
300	3	TRUE	4	0,627	0,599	0,634	0,617	0,583	0,488	0,398	0,443
300	1	FALSE	4	0,566	0,626	0,402	0,514	0,557	0,541	0,383	0,462
300	1	TRUE	4	0,585	0,651	0,575	0,613	0,557	0,541	0,383	0,462
300	7	FALSE	4	0,671	0,637	0,505	0,571	0,586	0,489	0,401	0,445
300	7	TRUE	4	0,664	0,650	0,668	0,659	0,584	0,506	0,402	0,454
300	12	FALSE	4	0,665	0,730	0,498	0,614	0,573	0,510	0,392	0,451
300	12	TRUE	4	0,689	0,750	0,669	0,709	0,563	0,534	0,387	0,460
300	26	FALSE	4	0,754	0,811	0,596	0,704	0,574	0,507	0,393	0,450
300	26	TRUE	4	0,773	0,840	0,741	0,790	0,563	0,522	0,385	0,453

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
300	38	FALSE	4	0,760	0,851	0,599	0,725	0,574	0,516	0,394	0,455
300	38	TRUE	4	0,790	0,884	0,744	0,814	0,565	0,534	0,389	0,461
300	3	FALSE	5	0,655	0,625	0,486	0,555	0,589	0,483	0,403	0,443
300	3	TRUE	5	0,648	0,655	0,645	0,650	0,582	0,524	0,402	0,463
300	1	FALSE	5	0,599	0,570	0,424	0,497	0,590	0,428	0,394	0,411
300	1	TRUE	5	0,595	0,660	0,584	0,622	0,587	0,478	0,400	0,439
300	7	FALSE	5	0,689	0,717	0,524	0,620	0,585	0,534	0,407	0,470
300	7	TRUE	5	0,694	0,737	0,678	0,708	0,581	0,558	0,407	0,482
300	12	FALSE	5	0,705	0,760	0,541	0,650	0,576	0,511	0,395	0,453
300	12	TRUE	5	0,724	0,829	0,685	0,757	0,550	0,577	0,384	0,480
300	26	FALSE	5	0,793	0,864	0,641	0,752	0,586	0,507	0,403	0,455
300	26	TRUE	5	0,813	0,879	0,777	0,828	0,581	0,519	0,401	0,460
300	38	FALSE	5	0,814	0,876	0,668	0,772	0,593	0,449	0,401	0,425
300	38	TRUE	5	0,832	0,882	0,801	0,841	0,592	0,461	0,402	0,432
300	3	FALSE	6	0,675	0,644	0,509	0,577	0,592	0,476	0,404	0,440
300	3	TRUE	6	0,663	0,660	0,663	0,662	0,586	0,508	0,404	0,456
300	1	FALSE	6	0,623	0,581	0,449	0,515	0,592	0,420	0,395	0,407
300	1	TRUE	6	0,613	0,622	0,611	0,616	0,574	0,493	0,390	0,441
300	7	FALSE	6	0,708	0,773	0,543	0,658	0,583	0,545	0,406	0,476
300	7	TRUE	6	0,717	0,756	0,701	0,728	0,580	0,549	0,404	0,477
300	12	FALSE	6	0,728	0,814	0,564	0,689	0,570	0,549	0,396	0,473
300	12	TRUE	6	0,752	0,822	0,721	0,772	0,574	0,543	0,398	0,470
300	26	FALSE	6	0,827	0,891	0,685	0,788	0,589	0,505	0,406	0,455
300	26	TRUE	6	0,844	0,894	0,813	0,853	0,588	0,497	0,404	0,450
300	38	FALSE	6	0,847	0,896	0,716	0,806	0,603	0,453	0,413	0,433
300	38	TRUE	6	0,861	0,904	0,833	0,868	0,606	0,441	0,414	0,427
300	3	FALSE	7	0,689	0,675	0,525	0,600	0,594	0,471	0,406	0,438
300	3	TRUE	7	0,680	0,715	0,668	0,691	0,582	0,527	0,403	0,465
300	1	FALSE	7	0,616	0,629	0,446	0,537	0,598	0,406	0,398	0,402
300	1	TRUE	7	0,624	0,669	0,613	0,641	0,580	0,478	0,393	0,435
300	7	FALSE	7	0,721	0,809	0,556	0,682	0,586	0,539	0,408	0,473
300	7	TRUE	7	0,734	0,781	0,713	0,747	0,584	0,546	0,408	0,477
300	12	FALSE	7	0,760	0,834	0,600	0,717	0,577	0,517	0,396	0,457
300	12	TRUE	7	0,778	0,833	0,750	0,791	0,579	0,514	0,398	0,456
300	26	FALSE	7	0,852	0,910	0,719	0,815	0,598	0,474	0,411	0,443
300	26	TRUE	7	0,868	0,916	0,836	0,876	0,591	0,475	0,404	0,439
300	38	FALSE	7	0,866	0,926	0,738	0,832	0,606	0,451	0,416	0,433
300	38	TRUE	7	0,883	0,927	0,851	0,889	0,610	0,447	0,421	0,434
300	3	FALSE	8	0,696	0,699	0,533	0,616	0,591	0,471	0,402	0,437
300	3	TRUE	8	0,695	0,731	0,682	0,707	0,588	0,512	0,407	0,459
300	1	FALSE	8	0,622	0,729	0,457	0,593	0,540	0,588	0,378	0,483
300	1	TRUE	8	0,634	0,670	0,624	0,647	0,580	0,478	0,393	0,435
300	7	FALSE	8	0,744	0,810	0,583	0,697	0,590	0,528	0,410	0,469
300	7	TRUE	8	0,750	0,787	0,732	0,760	0,589	0,531	0,410	0,471
300	12	FALSE	8	0,786	0,849	0,633	0,741	0,586	0,493	0,402	0,447
300	12	TRUE	8	0,795	0,829	0,776	0,803	0,589	0,484	0,403	0,443
300	26	FALSE	8	0,869	0,913	0,748	0,830	0,604	0,448	0,414	0,431
300	26	TRUE	8	0,883	0,919	0,856	0,888	0,599	0,456	0,409	0,432
300	38	FALSE	8	0,883	0,929	0,768	0,849	0,606	0,430	0,413	0,421
300	38	TRUE	8	0,898	0,933	0,872	0,903	0,611	0,428	0,418	0,423
300	3	FALSE	9	0,707	0,709	0,546	0,627	0,590	0,459	0,400	0,430
300	3	TRUE	9	0,704	0,739	0,690	0,714	0,589	0,509	0,407	0,458
300	1	FALSE	9	0,638	0,729	0,471	0,600	0,540	0,587	0,378	0,483
300	1	TRUE	9	0,645	0,647	0,644	0,645	0,583	0,462	0,393	0,428
300	7	FALSE	9	0,751	0,826	0,590	0,708	0,587	0,529	0,408	0,468
300	7	TRUE	9	0,763	0,801	0,744	0,773	0,590	0,526	0,410	0,468
300	12	FALSE	9	0,796	0,869	0,643	0,756	0,584	0,492	0,400	0,446
300	12	TRUE	9	0,810	0,850	0,786	0,818	0,590	0,488	0,405	0,446
300	26	FALSE	9	0,877	0,918	0,762	0,840	0,607	0,446	0,417	0,431
300	26	TRUE	9	0,890	0,922	0,867	0,894	0,601	0,450	0,411	0,431
300	38	FALSE	9	0,895	0,933	0,791	0,862	0,611	0,426	0,418	0,422
300	38	TRUE	9	0,909	0,935	0,888	0,912	0,614	0,424	0,422	0,423
300	3	FALSE	10	0,715	0,754	0,552	0,653	0,589	0,487	0,403	0,445
300	3	TRUE	10	0,709	0,728	0,701	0,714	0,598	0,490	0,413	0,451

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
300	1	FALSE	10	0,651	0,720	0,484	0,602	0,567	0,519	0,388	0,454
300	1	TRUE	10	0,649	0,693	0,636	0,665	0,544	0,581	0,380	0,481
300	7	FALSE	10	0,769	0,813	0,616	0,715	0,595	0,504	0,412	0,458
300	7	TRUE	10	0,769	0,803	0,752	0,777	0,591	0,520	0,411	0,465
300	12	FALSE	10	0,806	0,874	0,657	0,766	0,587	0,487	0,401	0,444
300	12	TRUE	10	0,818	0,854	0,796	0,825	0,591	0,480	0,404	0,442
300	26	FALSE	10	0,884	0,920	0,774	0,847	0,608	0,437	0,416	0,426
300	26	TRUE	10	0,894	0,924	0,871	0,898	0,603	0,447	0,412	0,430
300	38	FALSE	10	0,902	0,938	0,801	0,869	0,610	0,427	0,417	0,422
300	38	TRUE	10	0,914	0,941	0,893	0,917	0,614	0,422	0,422	0,422
500	3	FALSE	∞	0,742	0,789	0,583	0,686	0,588	0,469	0,398	0,433
500	3	TRUE	∞	0,722	0,751	0,709	0,730	0,594	0,482	0,407	0,444
500	1	FALSE	∞	0,687	0,675	0,522	0,599	0,590	0,425	0,393	0,409
500	1	TRUE	∞	0,653	0,666	0,649	0,658	0,564	0,511	0,383	0,447
500	7	FALSE	∞	0,766	0,825	0,609	0,717	0,595	0,493	0,409	0,451
500	7	TRUE	∞	0,766	0,801	0,748	0,774	0,594	0,510	0,411	0,461
500	12	FALSE	∞	0,828	0,864	0,693	0,779	0,608	0,455	0,418	0,436
500	12	TRUE	∞	0,822	0,845	0,807	0,826	0,604	0,446	0,412	0,429
500	26	FALSE	∞	0,886	0,919	0,779	0,849	0,616	0,428	0,423	0,425
500	26	TRUE	∞	0,890	0,914	0,872	0,893	0,618	0,427	0,426	0,426
500	38	FALSE	∞	0,902	0,932	0,804	0,868	0,613	0,415	0,417	0,416
500	38	TRUE	∞	0,914	0,938	0,895	0,916	0,615	0,402	0,418	0,410
500	3	FALSE	1	0,569	0,518	0,389	0,454	0,551	0,487	0,368	0,428
500	3	TRUE	1	0,556	0,531	0,558	0,545	0,546	0,503	0,367	0,435
500	1	FALSE	1	0,447	0,838	0,358	0,598	0,446	0,837	0,358	0,597
500	1	TRUE	1	0,545	0,839	0,528	0,683	0,446	0,837	0,358	0,597
500	7	FALSE	1	0,553	0,649	0,395	0,522	0,529	0,613	0,374	0,493
500	7	TRUE	1	0,577	0,649	0,566	0,608	0,529	0,613	0,373	0,493
500	12	FALSE	1	0,525	0,752	0,389	0,571	0,499	0,662	0,362	0,512
500	12	TRUE	1	0,581	0,759	0,559	0,659	0,494	0,672	0,360	0,516
500	26	FALSE	1	0,574	0,758	0,422	0,590	0,520	0,670	0,376	0,523
500	26	TRUE	1	0,620	0,761	0,593	0,677	0,520	0,675	0,376	0,526
500	38	FALSE	1	0,577	0,777	0,425	0,601	0,530	0,689	0,385	0,537
500	38	TRUE	1	0,628	0,773	0,598	0,686	0,530	0,678	0,383	0,531
500	3	FALSE	2	0,595	0,503	0,411	0,457	0,566	0,455	0,375	0,415
500	3	TRUE	2	0,575	0,507	0,586	0,547	0,571	0,456	0,380	0,418
500	1	FALSE	2	0,587	0,464	0,397	0,430	0,567	0,456	0,376	0,416
500	1	TRUE	2	0,556	0,685	0,543	0,614	0,530	0,562	0,365	0,464
500	7	FALSE	2	0,596	0,634	0,427	0,531	0,555	0,551	0,383	0,467
500	7	TRUE	2	0,607	0,633	0,601	0,617	0,558	0,547	0,384	0,466
500	12	FALSE	2	0,622	0,584	0,447	0,515	0,572	0,456	0,381	0,419
500	12	TRUE	2	0,617	0,654	0,608	0,631	0,558	0,505	0,377	0,441
500	26	FALSE	2	0,646	0,710	0,478	0,594	0,564	0,551	0,390	0,471
500	26	TRUE	2	0,668	0,761	0,640	0,701	0,550	0,602	0,387	0,495
500	38	FALSE	2	0,653	0,754	0,486	0,620	0,568	0,562	0,395	0,479
500	38	TRUE	2	0,681	0,780	0,651	0,715	0,560	0,594	0,393	0,493
500	3	FALSE	3	0,602	0,587	0,427	0,507	0,565	0,493	0,381	0,437
500	3	TRUE	3	0,600	0,632	0,593	0,612	0,556	0,540	0,382	0,461
500	1	FALSE	3	0,503	0,780	0,380	0,580	0,485	0,777	0,370	0,573
500	1	TRUE	3	0,571	0,791	0,548	0,669	0,482	0,772	0,367	0,570
500	7	FALSE	3	0,621	0,617	0,449	0,533	0,566	0,518	0,386	0,452
500	7	TRUE	3	0,623	0,640	0,618	0,629	0,564	0,522	0,385	0,453
500	12	FALSE	3	0,644	0,666	0,475	0,570	0,575	0,521	0,395	0,458
500	12	TRUE	3	0,657	0,741	0,634	0,687	0,544	0,593	0,381	0,487
500	26	FALSE	3	0,693	0,738	0,528	0,633	0,573	0,538	0,396	0,467
500	26	TRUE	3	0,713	0,807	0,678	0,743	0,547	0,592	0,383	0,488
500	38	FALSE	3	0,709	0,751	0,545	0,648	0,583	0,502	0,399	0,450
500	38	TRUE	3	0,731	0,836	0,690	0,763	0,553	0,587	0,387	0,487
500	3	FALSE	4	0,624	0,600	0,450	0,525	0,571	0,491	0,386	0,439
500	3	TRUE	4	0,621	0,733	0,598	0,666	0,536	0,622	0,380	0,501
500	1	FALSE	4	0,531	0,729	0,390	0,560	0,520	0,662	0,374	0,518
500	1	TRUE	4	0,579	0,745	0,559	0,652	0,508	0,689	0,371	0,530
500	7	FALSE	4	0,655	0,656	0,486	0,571	0,570	0,517	0,390	0,454
500	7	TRUE	4	0,661	0,666	0,658	0,662	0,568	0,522	0,389	0,455

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
500	12	FALSE	4	0,684	0,680	0,518	0,599	0,579	0,483	0,392	0,437
500	12	TRUE	4	0,690	0,728	0,676	0,702	0,570	0,522	0,390	0,456
500	26	FALSE	4	0,746	0,824	0,584	0,704	0,578	0,530	0,399	0,464
500	26	TRUE	4	0,768	0,840	0,733	0,787	0,573	0,537	0,395	0,466
500	38	FALSE	4	0,756	0,824	0,597	0,710	0,589	0,481	0,402	0,442
500	38	TRUE	4	0,783	0,866	0,741	0,804	0,582	0,520	0,401	0,460
500	3	FALSE	5	0,649	0,600	0,478	0,539	0,587	0,448	0,393	0,421
500	3	TRUE	5	0,643	0,691	0,630	0,660	0,568	0,546	0,393	0,469
500	1	FALSE	5	0,601	0,570	0,425	0,498	0,600	0,396	0,398	0,397
500	1	TRUE	5	0,592	0,644	0,582	0,613	0,550	0,541	0,377	0,459
500	7	FALSE	5	0,693	0,650	0,531	0,590	0,598	0,445	0,406	0,425
500	7	TRUE	5	0,686	0,686	0,685	0,685	0,591	0,492	0,405	0,449
500	12	FALSE	5	0,717	0,742	0,556	0,649	0,585	0,500	0,400	0,450
500	12	TRUE	5	0,725	0,755	0,711	0,733	0,579	0,508	0,396	0,452
500	26	FALSE	5	0,786	0,863	0,630	0,746	0,585	0,514	0,403	0,459
500	26	TRUE	5	0,807	0,873	0,770	0,822	0,584	0,520	0,403	0,461
500	38	FALSE	5	0,800	0,883	0,646	0,765	0,595	0,500	0,411	0,456
500	38	TRUE	5	0,825	0,905	0,780	0,843	0,590	0,518	0,409	0,463
500	3	FALSE	6	0,675	0,613	0,509	0,561	0,591	0,436	0,395	0,416
500	3	TRUE	6	0,662	0,684	0,654	0,669	0,577	0,498	0,393	0,445
500	1	FALSE	6	0,606	0,627	0,435	0,531	0,597	0,407	0,396	0,402
500	1	TRUE	6	0,600	0,672	0,587	0,629	0,550	0,539	0,376	0,458
500	7	FALSE	6	0,704	0,737	0,540	0,638	0,578	0,515	0,397	0,456
500	7	TRUE	6	0,711	0,735	0,700	0,718	0,587	0,513	0,404	0,458
500	12	FALSE	6	0,749	0,784	0,593	0,688	0,597	0,490	0,411	0,450
500	12	TRUE	6	0,755	0,774	0,744	0,759	0,598	0,481	0,411	0,446
500	26	FALSE	6	0,820	0,878	0,676	0,777	0,601	0,486	0,415	0,450
500	26	TRUE	6	0,835	0,875	0,810	0,842	0,603	0,478	0,415	0,447
500	38	FALSE	6	0,840	0,899	0,702	0,801	0,608	0,458	0,419	0,438
500	38	TRUE	6	0,858	0,907	0,826	0,866	0,608	0,464	0,420	0,442
500	3	FALSE	7	0,686	0,652	0,522	0,587	0,591	0,457	0,400	0,428
500	3	TRUE	7	0,678	0,705	0,667	0,686	0,581	0,501	0,398	0,449
500	1	FALSE	7	0,623	0,625	0,451	0,538	0,597	0,406	0,397	0,402
500	1	TRUE	7	0,609	0,680	0,595	0,637	0,550	0,539	0,377	0,458
500	7	FALSE	7	0,718	0,769	0,555	0,662	0,582	0,521	0,402	0,461
500	7	TRUE	7	0,727	0,772	0,708	0,740	0,581	0,532	0,402	0,467
500	12	FALSE	7	0,773	0,821	0,620	0,720	0,594	0,484	0,407	0,446
500	12	TRUE	7	0,779	0,811	0,762	0,786	0,596	0,473	0,408	0,441
500	26	FALSE	7	0,841	0,901	0,703	0,802	0,608	0,474	0,421	0,447
500	26	TRUE	7	0,856	0,894	0,831	0,862	0,610	0,459	0,421	0,440
500	38	FALSE	7	0,861	0,917	0,732	0,824	0,613	0,445	0,422	0,434
500	38	TRUE	7	0,879	0,920	0,850	0,885	0,611	0,430	0,418	0,424
500	3	FALSE	8	0,704	0,710	0,542	0,626	0,587	0,476	0,398	0,437
500	3	TRUE	8	0,692	0,697	0,689	0,693	0,587	0,462	0,397	0,429
500	1	FALSE	8	0,646	0,619	0,475	0,547	0,592	0,419	0,394	0,406
500	1	TRUE	8	0,617	0,687	0,602	0,645	0,548	0,546	0,377	0,461
500	7	FALSE	8	0,743	0,761	0,587	0,674	0,593	0,494	0,407	0,451
500	7	TRUE	8	0,745	0,756	0,739	0,748	0,595	0,500	0,410	0,455
500	12	FALSE	8	0,793	0,839	0,645	0,742	0,602	0,476	0,414	0,445
500	12	TRUE	8	0,795	0,825	0,778	0,801	0,601	0,468	0,412	0,440
500	26	FALSE	8	0,859	0,908	0,732	0,820	0,613	0,451	0,424	0,437
500	26	TRUE	8	0,871	0,902	0,848	0,875	0,614	0,449	0,425	0,437
500	38	FALSE	8	0,874	0,925	0,752	0,838	0,610	0,433	0,417	0,425
500	38	TRUE	8	0,891	0,933	0,860	0,896	0,610	0,424	0,416	0,420
500	3	FALSE	9	0,704	0,771	0,538	0,654	0,574	0,501	0,391	0,446
500	3	TRUE	9	0,701	0,731	0,689	0,710	0,585	0,485	0,398	0,441
500	1	FALSE	9	0,651	0,663	0,482	0,572	0,562	0,498	0,380	0,439
500	1	TRUE	9	0,631	0,628	0,630	0,629	0,598	0,401	0,397	0,399
500	7	FALSE	9	0,750	0,786	0,593	0,690	0,592	0,504	0,409	0,456
500	7	TRUE	9	0,754	0,791	0,735	0,763	0,588	0,522	0,407	0,465
500	12	FALSE	9	0,803	0,853	0,657	0,755	0,604	0,470	0,416	0,443
500	12	TRUE	9	0,803	0,834	0,785	0,810	0,601	0,464	0,412	0,438
500	26	FALSE	9	0,870	0,912	0,751	0,831	0,614	0,449	0,424	0,437
500	26	TRUE	9	0,878	0,904	0,859	0,881	0,616	0,438	0,426	0,432

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
500	38	FALSE	9	0,886	0,924	0,775	0,850	0,610	0,422	0,415	0,419
500	38	TRUE	9	0,902	0,935	0,877	0,906	0,612	0,415	0,417	0,416
500	3	FALSE	10	0,719	0,768	0,556	0,662	0,588	0,486	0,402	0,444
500	3	TRUE	10	0,711	0,727	0,703	0,715	0,595	0,465	0,405	0,435
500	1	FALSE	10	0,665	0,653	0,497	0,575	0,594	0,406	0,393	0,399
500	1	TRUE	10	0,641	0,637	0,640	0,639	0,598	0,399	0,396	0,398
500	7	FALSE	10	0,752	0,804	0,594	0,699	0,593	0,514	0,411	0,463
500	7	TRUE	10	0,760	0,800	0,740	0,770	0,591	0,526	0,411	0,468
500	12	FALSE	10	0,814	0,858	0,673	0,765	0,607	0,463	0,418	0,441
500	12	TRUE	10	0,812	0,840	0,794	0,817	0,603	0,457	0,413	0,435
500	26	FALSE	10	0,877	0,917	0,761	0,839	0,616	0,444	0,425	0,435
500	26	TRUE	10	0,885	0,913	0,864	0,888	0,616	0,440	0,425	0,433
500	38	FALSE	10	0,893	0,931	0,786	0,859	0,613	0,423	0,419	0,421
500	38	TRUE	10	0,908	0,937	0,886	0,911	0,614	0,407	0,418	0,413
750	3	FALSE	∞	0,723	0,749	0,563	0,656	0,599	0,501	0,416	0,459
750	3	TRUE	∞	0,714	0,749	0,699	0,724	0,591	0,522	0,410	0,466
750	1	FALSE	∞	0,675	0,723	0,508	0,615	0,563	0,519	0,385	0,452
750	1	TRUE	∞	0,654	0,683	0,645	0,664	0,528	0,621	0,375	0,498
750	7	FALSE	∞	0,767	0,792	0,617	0,704	0,597	0,487	0,411	0,449
750	7	TRUE	∞	0,766	0,808	0,745	0,777	0,590	0,507	0,407	0,457
750	12	FALSE	∞	0,820	0,853	0,684	0,769	0,613	0,436	0,422	0,429
750	12	TRUE	∞	0,823	0,848	0,807	0,827	0,612	0,451	0,423	0,437
750	26	FALSE	∞	0,877	0,918	0,761	0,839	0,617	0,408	0,423	0,415
750	26	TRUE	∞	0,891	0,917	0,872	0,894	0,621	0,394	0,425	0,410
750	38	FALSE	∞	0,903	0,933	0,806	0,869	0,620	0,400	0,425	0,413
750	38	TRUE	∞	0,910	0,931	0,894	0,913	0,620	0,387	0,424	0,405
750	3	FALSE	1	0,573	0,510	0,391	0,450	0,553	0,485	0,370	0,427
750	3	TRUE	1	0,557	0,505	0,563	0,534	0,555	0,481	0,371	0,426
750	1	FALSE	1	0,455	0,821	0,360	0,591	0,458	0,802	0,360	0,581
750	1	TRUE	1	0,547	0,823	0,530	0,676	0,458	0,802	0,360	0,581
750	7	FALSE	1	0,569	0,600	0,401	0,501	0,546	0,580	0,381	0,481
750	7	TRUE	1	0,577	0,601	0,573	0,587	0,546	0,580	0,381	0,480
750	12	FALSE	1	0,516	0,781	0,388	0,584	0,492	0,691	0,363	0,527
750	12	TRUE	1	0,582	0,787	0,558	0,672	0,488	0,701	0,362	0,531
750	26	FALSE	1	0,570	0,769	0,421	0,595	0,519	0,684	0,378	0,531
750	26	TRUE	1	0,620	0,768	0,593	0,680	0,520	0,680	0,377	0,529
750	38	FALSE	1	0,579	0,776	0,427	0,602	0,533	0,702	0,389	0,545
750	38	TRUE	1	0,629	0,773	0,600	0,687	0,539	0,693	0,392	0,542
750	3	FALSE	2	0,606	0,472	0,418	0,445	0,578	0,413	0,378	0,396
750	3	TRUE	2	0,572	0,468	0,590	0,529	0,579	0,411	0,379	0,395
750	1	FALSE	2	0,519	0,677	0,377	0,527	0,506	0,614	0,359	0,487
750	1	TRUE	2	0,559	0,678	0,548	0,613	0,506	0,614	0,359	0,487
750	7	FALSE	2	0,608	0,601	0,436	0,518	0,568	0,526	0,390	0,458
750	7	TRUE	2	0,607	0,605	0,607	0,606	0,567	0,529	0,390	0,459
750	12	FALSE	2	0,621	0,565	0,446	0,505	0,580	0,439	0,386	0,413
750	12	TRUE	2	0,612	0,629	0,608	0,618	0,565	0,483	0,380	0,432
750	26	FALSE	2	0,647	0,713	0,479	0,596	0,567	0,555	0,394	0,474
750	26	TRUE	2	0,668	0,749	0,644	0,697	0,559	0,579	0,391	0,485
750	38	FALSE	2	0,655	0,739	0,488	0,613	0,581	0,553	0,406	0,479
750	38	TRUE	2	0,680	0,757	0,655	0,706	0,579	0,569	0,406	0,487
750	3	FALSE	3	0,573	0,663	0,412	0,538	0,541	0,572	0,376	0,474
750	3	TRUE	3	0,600	0,716	0,581	0,648	0,526	0,623	0,373	0,498
750	1	FALSE	3	0,508	0,779	0,382	0,581	0,486	0,752	0,367	0,560
750	1	TRUE	3	0,575	0,780	0,553	0,667	0,456	0,822	0,361	0,591
750	7	FALSE	3	0,632	0,615	0,460	0,538	0,576	0,503	0,393	0,448
750	7	TRUE	3	0,629	0,631	0,628	0,629	0,570	0,507	0,389	0,448
750	12	FALSE	3	0,633	0,673	0,465	0,569	0,570	0,533	0,393	0,463
750	12	TRUE	3	0,656	0,769	0,627	0,698	0,546	0,603	0,385	0,494
750	26	FALSE	3	0,697	0,739	0,532	0,636	0,579	0,511	0,398	0,454
750	26	TRUE	3	0,716	0,816	0,680	0,748	0,550	0,591	0,386	0,488
750	38	FALSE	3	0,707	0,767	0,542	0,655	0,581	0,517	0,400	0,459
750	38	TRUE	3	0,730	0,848	0,686	0,767	0,559	0,580	0,391	0,486
750	3	FALSE	4	0,607	0,661	0,440	0,550	0,560	0,538	0,385	0,461
750	3	TRUE	4	0,623	0,742	0,599	0,670	0,539	0,628	0,383	0,506

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
750	1	FALSE	4	0,541	0,707	0,395	0,551	0,531	0,622	0,376	0,499
750	1	TRUE	4	0,581	0,716	0,563	0,640	0,494	0,705	0,365	0,535
750	7	FALSE	4	0,648	0,679	0,480	0,579	0,560	0,542	0,386	0,464
750	7	TRUE	4	0,660	0,707	0,646	0,677	0,557	0,564	0,387	0,476
750	12	FALSE	4	0,691	0,687	0,528	0,608	0,598	0,479	0,412	0,446
750	12	TRUE	4	0,697	0,730	0,684	0,707	0,586	0,517	0,405	0,461
750	26	FALSE	4	0,736	0,820	0,573	0,696	0,580	0,532	0,401	0,467
750	26	TRUE	4	0,762	0,833	0,730	0,781	0,579	0,532	0,401	0,466
750	38	FALSE	4	0,762	0,811	0,606	0,709	0,601	0,495	0,417	0,456
750	38	TRUE	4	0,783	0,845	0,752	0,798	0,599	0,500	0,415	0,458
750	3	FALSE	5	0,641	0,670	0,472	0,571	0,579	0,534	0,401	0,467
750	3	TRUE	5	0,649	0,684	0,639	0,662	0,576	0,546	0,400	0,473
750	1	FALSE	5	0,569	0,681	0,411	0,546	0,545	0,550	0,375	0,463
750	1	TRUE	5	0,591	0,694	0,576	0,635	0,510	0,628	0,364	0,496
750	7	FALSE	5	0,700	0,621	0,544	0,583	0,603	0,434	0,410	0,422
750	7	TRUE	5	0,685	0,661	0,694	0,677	0,594	0,463	0,404	0,434
750	12	FALSE	5	0,723	0,727	0,565	0,646	0,608	0,486	0,423	0,454
750	12	TRUE	5	0,731	0,782	0,709	0,745	0,591	0,519	0,411	0,465
750	26	FALSE	5	0,776	0,862	0,617	0,739	0,588	0,508	0,406	0,457
750	26	TRUE	5	0,801	0,871	0,764	0,818	0,585	0,512	0,403	0,457
750	38	FALSE	5	0,807	0,868	0,659	0,764	0,607	0,470	0,419	0,445
750	38	TRUE	5	0,826	0,885	0,792	0,838	0,604	0,477	0,417	0,447
750	3	FALSE	6	0,661	0,677	0,493	0,585	0,583	0,519	0,403	0,461
750	3	TRUE	6	0,666	0,687	0,659	0,673	0,580	0,524	0,401	0,463
750	1	FALSE	6	0,605	0,634	0,436	0,535	0,543	0,609	0,383	0,496
750	1	TRUE	6	0,607	0,647	0,598	0,623	0,507	0,695	0,372	0,533
750	7	FALSE	6	0,711	0,698	0,552	0,625	0,588	0,492	0,403	0,447
750	7	TRUE	6	0,710	0,714	0,707	0,711	0,586	0,491	0,401	0,446
750	12	FALSE	6	0,745	0,804	0,586	0,695	0,603	0,476	0,416	0,446
750	12	TRUE	6	0,761	0,811	0,737	0,774	0,601	0,487	0,416	0,452
750	26	FALSE	6	0,811	0,868	0,666	0,767	0,606	0,483	0,420	0,451
750	26	TRUE	6	0,830	0,887	0,796	0,841	0,602	0,487	0,416	0,452
750	38	FALSE	6	0,834	0,900	0,692	0,796	0,610	0,470	0,424	0,447
750	38	TRUE	6	0,854	0,906	0,821	0,863	0,611	0,449	0,422	0,436
750	3	FALSE	7	0,682	0,703	0,516	0,610	0,590	0,520	0,410	0,465
750	3	TRUE	7	0,684	0,715	0,672	0,694	0,582	0,534	0,404	0,469
750	1	FALSE	7	0,601	0,681	0,436	0,559	0,545	0,598	0,383	0,490
750	1	TRUE	7	0,613	0,680	0,600	0,640	0,507	0,687	0,371	0,529
750	7	FALSE	7	0,721	0,756	0,560	0,658	0,589	0,512	0,407	0,460
750	7	TRUE	7	0,727	0,746	0,719	0,732	0,597	0,480	0,410	0,445
750	12	FALSE	7	0,771	0,834	0,615	0,724	0,601	0,502	0,418	0,460
750	12	TRUE	7	0,783	0,830	0,759	0,794	0,598	0,509	0,416	0,462
750	26	FALSE	7	0,833	0,900	0,692	0,796	0,610	0,475	0,424	0,449
750	26	TRUE	7	0,853	0,902	0,822	0,862	0,610	0,463	0,422	0,443
750	38	FALSE	7	0,858	0,906	0,731	0,819	0,612	0,427	0,420	0,423
750	38	TRUE	7	0,873	0,910	0,847	0,878	0,609	0,408	0,413	0,410
750	3	FALSE	8	0,696	0,712	0,532	0,622	0,595	0,509	0,413	0,461
750	3	TRUE	8	0,696	0,735	0,682	0,708	0,585	0,528	0,406	0,467
750	1	FALSE	8	0,619	0,713	0,454	0,583	0,542	0,584	0,378	0,481
750	1	TRUE	8	0,627	0,717	0,608	0,662	0,503	0,693	0,369	0,531
750	7	FALSE	8	0,742	0,759	0,588	0,673	0,593	0,499	0,410	0,454
750	7	TRUE	8	0,745	0,777	0,730	0,754	0,586	0,506	0,404	0,455
750	12	FALSE	8	0,791	0,843	0,642	0,742	0,606	0,475	0,419	0,447
750	12	TRUE	8	0,800	0,840	0,778	0,809	0,605	0,487	0,421	0,454
750	26	FALSE	8	0,848	0,907	0,714	0,810	0,611	0,453	0,422	0,438
750	26	TRUE	8	0,867	0,909	0,838	0,874	0,614	0,437	0,423	0,430
750	38	FALSE	8	0,872	0,924	0,750	0,837	0,614	0,423	0,421	0,422
750	38	TRUE	8	0,888	0,920	0,864	0,892	0,614	0,398	0,417	0,407
750	3	FALSE	9	0,707	0,726	0,545	0,635	0,597	0,504	0,414	0,459
750	3	TRUE	9	0,706	0,733	0,695	0,714	0,591	0,511	0,409	0,460
750	1	FALSE	9	0,635	0,698	0,468	0,583	0,543	0,588	0,380	0,484
750	1	TRUE	9	0,633	0,689	0,619	0,654	0,503	0,692	0,369	0,530
750	7	FALSE	9	0,744	0,784	0,587	0,686	0,589	0,505	0,406	0,456
750	7	TRUE	9	0,752	0,783	0,737	0,760	0,590	0,495	0,406	0,450

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
750	12	FALSE	9	0,804	0,850	0,660	0,755	0,609	0,457	0,420	0,438
750	12	TRUE	9	0,811	0,847	0,790	0,818	0,606	0,471	0,419	0,445
750	26	FALSE	9	0,859	0,918	0,729	0,823	0,612	0,442	0,422	0,432
750	26	TRUE	9	0,878	0,918	0,850	0,884	0,615	0,422	0,423	0,422
750	38	FALSE	9	0,886	0,929	0,775	0,852	0,616	0,415	0,423	0,419
750	38	TRUE	9	0,900	0,927	0,878	0,903	0,616	0,393	0,419	0,406
750	3	FALSE	10	0,713	0,741	0,551	0,646	0,595	0,513	0,413	0,463
750	3	TRUE	10	0,709	0,739	0,697	0,718	0,589	0,520	0,409	0,464
750	1	FALSE	10	0,650	0,685	0,482	0,583	0,552	0,567	0,384	0,475
750	1	TRUE	10	0,639	0,667	0,631	0,649	0,508	0,685	0,371	0,528
750	7	FALSE	10	0,754	0,783	0,600	0,691	0,594	0,496	0,410	0,453
750	7	TRUE	10	0,757	0,787	0,743	0,765	0,592	0,494	0,408	0,451
750	12	FALSE	10	0,812	0,854	0,672	0,763	0,611	0,453	0,422	0,437
750	12	TRUE	10	0,817	0,848	0,798	0,823	0,609	0,466	0,421	0,444
750	26	FALSE	10	0,867	0,916	0,744	0,830	0,615	0,422	0,422	0,422
750	26	TRUE	10	0,884	0,915	0,861	0,888	0,618	0,407	0,424	0,416
750	38	FALSE	10	0,894	0,930	0,788	0,859	0,616	0,406	0,421	0,413
750	38	TRUE	10	0,905	0,929	0,887	0,908	0,617	0,386	0,419	0,403
1000	3	FALSE	∞	0,717	0,753	0,555	0,654	0,591	0,501	0,407	0,454
1000	3	TRUE	∞	0,718	0,762	0,699	0,731	0,583	0,517	0,402	0,460
1000	1	FALSE	∞	0,689	0,687	0,525	0,606	0,581	0,469	0,392	0,431
1000	1	TRUE	∞	0,654	0,707	0,638	0,673	0,544	0,586	0,380	0,483
1000	7	FALSE	∞	0,757	0,793	0,602	0,698	0,593	0,495	0,408	0,452
1000	7	TRUE	∞	0,759	0,783	0,746	0,765	0,591	0,489	0,405	0,447
1000	12	FALSE	∞	0,816	0,866	0,674	0,770	0,616	0,434	0,425	0,430
1000	12	TRUE	∞	0,822	0,854	0,803	0,828	0,620	0,427	0,429	0,428
1000	26	FALSE	∞	0,877	0,914	0,764	0,839	0,617	0,410	0,422	0,416
1000	26	TRUE	∞	0,889	0,914	0,870	0,892	0,620	0,407	0,426	0,417
1000	38	FALSE	∞	0,903	0,928	0,808	0,868	0,623	0,397	0,429	0,413
1000	38	TRUE	∞	0,909	0,931	0,892	0,911	0,621	0,398	0,426	0,412
1000	3	FALSE	1	0,571	0,514	0,390	0,452	0,553	0,482	0,369	0,426
1000	3	TRUE	1	0,557	0,514	0,561	0,538	0,553	0,482	0,369	0,426
1000	1	FALSE	1	0,445	0,842	0,358	0,600	0,444	0,838	0,357	0,598
1000	1	TRUE	1	0,547	0,822	0,530	0,676	0,457	0,802	0,359	0,580
1000	7	FALSE	1	0,562	0,607	0,397	0,502	0,539	0,579	0,375	0,477
1000	7	TRUE	1	0,573	0,607	0,568	0,588	0,539	0,579	0,375	0,477
1000	12	FALSE	1	0,521	0,763	0,388	0,575	0,500	0,666	0,363	0,514
1000	12	TRUE	1	0,581	0,734	0,562	0,648	0,509	0,636	0,364	0,500
1000	26	FALSE	1	0,570	0,771	0,420	0,595	0,522	0,677	0,378	0,528
1000	26	TRUE	1	0,620	0,769	0,592	0,681	0,524	0,675	0,379	0,527
1000	38	FALSE	1	0,583	0,771	0,430	0,600	0,542	0,688	0,393	0,540
1000	38	TRUE	1	0,631	0,783	0,600	0,691	0,538	0,703	0,392	0,548
1000	3	FALSE	2	0,600	0,472	0,412	0,442	0,577	0,426	0,380	0,403
1000	3	TRUE	2	0,568	0,472	0,584	0,528	0,577	0,426	0,379	0,403
1000	1	FALSE	2	0,586	0,466	0,396	0,431	0,566	0,456	0,375	0,415
1000	1	TRUE	2	0,560	0,661	0,549	0,605	0,517	0,578	0,360	0,469
1000	7	FALSE	2	0,608	0,585	0,433	0,509	0,565	0,505	0,383	0,444
1000	7	TRUE	2	0,605	0,601	0,605	0,603	0,564	0,519	0,385	0,452
1000	12	FALSE	2	0,621	0,568	0,446	0,507	0,577	0,435	0,381	0,408
1000	12	TRUE	2	0,615	0,627	0,611	0,619	0,569	0,469	0,381	0,425
1000	26	FALSE	2	0,645	0,733	0,478	0,605	0,567	0,564	0,394	0,479
1000	26	TRUE	2	0,671	0,768	0,643	0,705	0,557	0,591	0,391	0,491
1000	38	FALSE	2	0,664	0,712	0,497	0,604	0,595	0,522	0,414	0,468
1000	38	TRUE	2	0,682	0,768	0,655	0,711	0,575	0,557	0,401	0,479
1000	3	FALSE	3	0,572	0,656	0,410	0,533	0,540	0,573	0,375	0,474
1000	3	TRUE	3	0,596	0,708	0,578	0,643	0,524	0,624	0,372	0,498
1000	1	FALSE	3	0,502	0,784	0,380	0,582	0,483	0,780	0,369	0,575
1000	1	TRUE	3	0,577	0,771	0,555	0,663	0,458	0,828	0,363	0,595
1000	7	FALSE	3	0,617	0,651	0,448	0,549	0,555	0,549	0,383	0,466
1000	7	TRUE	3	0,627	0,664	0,617	0,641	0,551	0,566	0,382	0,474
1000	12	FALSE	3	0,649	0,622	0,479	0,550	0,582	0,488	0,396	0,442
1000	12	TRUE	3	0,659	0,734	0,638	0,686	0,559	0,568	0,389	0,478
1000	26	FALSE	3	0,694	0,751	0,528	0,640	0,574	0,521	0,395	0,458
1000	26	TRUE	3	0,717	0,804	0,685	0,744	0,561	0,571	0,391	0,481

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
1000	38	FALSE	3	0,715	0,747	0,553	0,650	0,593	0,513	0,411	0,462
1000	38	TRUE	3	0,736	0,842	0,695	0,768	0,571	0,588	0,401	0,494
1000	3	FALSE	4	0,613	0,637	0,443	0,540	0,567	0,523	0,388	0,456
1000	3	TRUE	4	0,620	0,670	0,608	0,639	0,558	0,549	0,385	0,467
1000	1	FALSE	4	0,536	0,714	0,392	0,553	0,525	0,639	0,375	0,507
1000	1	TRUE	4	0,584	0,716	0,566	0,641	0,500	0,686	0,366	0,526
1000	7	FALSE	4	0,646	0,666	0,477	0,571	0,566	0,524	0,387	0,456
1000	7	TRUE	4	0,655	0,696	0,643	0,669	0,559	0,551	0,387	0,469
1000	12	FALSE	4	0,693	0,656	0,531	0,594	0,603	0,446	0,411	0,428
1000	12	TRUE	4	0,695	0,737	0,679	0,708	0,580	0,516	0,399	0,457
1000	26	FALSE	4	0,740	0,801	0,579	0,690	0,581	0,520	0,401	0,461
1000	26	TRUE	4	0,762	0,833	0,729	0,781	0,576	0,534	0,398	0,466
1000	38	FALSE	4	0,761	0,809	0,606	0,707	0,604	0,484	0,418	0,451
1000	38	TRUE	4	0,787	0,864	0,747	0,806	0,602	0,512	0,420	0,466
1000	3	FALSE	5	0,650	0,605	0,479	0,542	0,590	0,466	0,401	0,433
1000	3	TRUE	5	0,643	0,650	0,640	0,645	0,581	0,508	0,399	0,453
1000	1	FALSE	5	0,595	0,598	0,423	0,510	0,579	0,489	0,393	0,441
1000	1	TRUE	5	0,599	0,665	0,587	0,626	0,535	0,584	0,373	0,478
1000	7	FALSE	5	0,694	0,639	0,534	0,587	0,592	0,456	0,401	0,429
1000	7	TRUE	5	0,683	0,666	0,689	0,678	0,584	0,500	0,400	0,450
1000	12	FALSE	5	0,723	0,727	0,564	0,646	0,605	0,483	0,419	0,451
1000	12	TRUE	5	0,731	0,755	0,719	0,737	0,606	0,494	0,422	0,458
1000	26	FALSE	5	0,774	0,866	0,614	0,740	0,587	0,514	0,405	0,460
1000	26	TRUE	5	0,799	0,876	0,758	0,817	0,586	0,521	0,405	0,463
1000	38	FALSE	5	0,805	0,865	0,657	0,761	0,603	0,481	0,417	0,449
1000	38	TRUE	5	0,826	0,889	0,789	0,839	0,599	0,496	0,415	0,455
1000	3	FALSE	6	0,661	0,635	0,493	0,564	0,589	0,477	0,401	0,439
1000	3	TRUE	6	0,662	0,682	0,655	0,668	0,579	0,517	0,398	0,457
1000	1	FALSE	6	0,582	0,682	0,421	0,551	0,578	0,490	0,393	0,441
1000	1	TRUE	6	0,611	0,587	0,616	0,602	0,549	0,561	0,380	0,471
1000	7	FALSE	6	0,700	0,721	0,537	0,629	0,575	0,518	0,395	0,457
1000	7	TRUE	6	0,703	0,716	0,697	0,706	0,577	0,517	0,396	0,457
1000	12	FALSE	6	0,744	0,737	0,593	0,665	0,611	0,445	0,420	0,432
1000	12	TRUE	6	0,756	0,812	0,730	0,771	0,602	0,494	0,417	0,456
1000	26	FALSE	6	0,808	0,881	0,658	0,769	0,599	0,475	0,412	0,443
1000	26	TRUE	6	0,829	0,887	0,795	0,841	0,601	0,482	0,414	0,448
1000	38	FALSE	6	0,837	0,904	0,696	0,800	0,612	0,452	0,422	0,437
1000	38	TRUE	6	0,856	0,903	0,825	0,864	0,612	0,453	0,423	0,438
1000	3	FALSE	7	0,683	0,661	0,518	0,589	0,597	0,477	0,409	0,443
1000	3	TRUE	7	0,685	0,722	0,671	0,696	0,582	0,522	0,402	0,462
1000	1	FALSE	7	0,603	0,687	0,438	0,563	0,546	0,592	0,383	0,487
1000	1	TRUE	7	0,621	0,696	0,605	0,650	0,512	0,688	0,374	0,531
1000	7	FALSE	7	0,709	0,762	0,545	0,654	0,581	0,524	0,401	0,463
1000	7	TRUE	7	0,718	0,745	0,707	0,726	0,586	0,521	0,405	0,463
1000	12	FALSE	7	0,755	0,850	0,592	0,721	0,594	0,504	0,410	0,457
1000	12	TRUE	7	0,780	0,845	0,747	0,796	0,604	0,497	0,419	0,458
1000	26	FALSE	7	0,833	0,896	0,692	0,794	0,607	0,454	0,417	0,435
1000	26	TRUE	7	0,852	0,897	0,822	0,860	0,611	0,454	0,422	0,438
1000	38	FALSE	7	0,859	0,910	0,732	0,821	0,613	0,429	0,421	0,425
1000	38	TRUE	7	0,875	0,911	0,850	0,880	0,615	0,428	0,423	0,426
1000	3	FALSE	8	0,696	0,671	0,534	0,603	0,597	0,462	0,408	0,435
1000	3	TRUE	8	0,697	0,726	0,686	0,706	0,589	0,509	0,406	0,457
1000	1	FALSE	8	0,625	0,676	0,457	0,566	0,577	0,487	0,391	0,439
1000	1	TRUE	8	0,633	0,756	0,606	0,681	0,508	0,703	0,373	0,538
1000	7	FALSE	8	0,735	0,764	0,577	0,670	0,587	0,508	0,404	0,456
1000	7	TRUE	8	0,734	0,760	0,722	0,741	0,583	0,500	0,399	0,450
1000	12	FALSE	8	0,776	0,850	0,619	0,735	0,603	0,479	0,416	0,448
1000	12	TRUE	8	0,793	0,849	0,763	0,806	0,609	0,480	0,423	0,452
1000	26	FALSE	8	0,849	0,908	0,715	0,811	0,609	0,436	0,416	0,426
1000	26	TRUE	8	0,865	0,909	0,835	0,872	0,612	0,436	0,421	0,428
1000	38	FALSE	8	0,875	0,918	0,757	0,838	0,618	0,420	0,425	0,422
1000	38	TRUE	8	0,889	0,928	0,860	0,894	0,613	0,432	0,421	0,426
1000	3	FALSE	9	0,699	0,696	0,537	0,617	0,593	0,473	0,405	0,439
1000	3	TRUE	9	0,710	0,746	0,695	0,721	0,586	0,515	0,405	0,460

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
1000	1	FALSE	9	0,637	0,722	0,470	0,596	0,578	0,486	0,392	0,439
1000	1	TRUE	9	0,643	0,709	0,626	0,668	0,543	0,591	0,380	0,486
1000	7	FALSE	9	0,739	0,779	0,580	0,679	0,590	0,510	0,407	0,459
1000	7	TRUE	9	0,743	0,763	0,733	0,748	0,590	0,493	0,405	0,449
1000	12	FALSE	9	0,787	0,857	0,632	0,745	0,607	0,472	0,420	0,446
1000	12	TRUE	9	0,803	0,850	0,776	0,813	0,614	0,465	0,427	0,446
1000	26	FALSE	9	0,861	0,916	0,733	0,825	0,611	0,430	0,418	0,424
1000	26	TRUE	9	0,877	0,916	0,850	0,883	0,616	0,431	0,424	0,428
1000	38	FALSE	9	0,885	0,929	0,772	0,850	0,617	0,419	0,423	0,421
1000	38	TRUE	9	0,898	0,930	0,874	0,902	0,615	0,421	0,422	0,421
1000	3	FALSE	10	0,702	0,742	0,538	0,640	0,586	0,505	0,403	0,454
1000	3	TRUE	10	0,714	0,760	0,695	0,728	0,584	0,520	0,403	0,462
1000	1	FALSE	10	0,665	0,686	0,497	0,592	0,581	0,469	0,392	0,430
1000	1	TRUE	10	0,646	0,701	0,631	0,666	0,544	0,586	0,380	0,483
1000	7	FALSE	10	0,744	0,789	0,586	0,688	0,591	0,504	0,408	0,456
1000	7	TRUE	10	0,751	0,778	0,737	0,758	0,589	0,499	0,405	0,452
1000	12	FALSE	10	0,800	0,860	0,651	0,755	0,612	0,446	0,421	0,434
1000	12	TRUE	10	0,813	0,856	0,787	0,822	0,615	0,442	0,425	0,434
1000	26	FALSE	10	0,872	0,908	0,756	0,832	0,614	0,409	0,419	0,414
1000	26	TRUE	10	0,883	0,908	0,865	0,886	0,619	0,410	0,425	0,417
1000	38	FALSE	10	0,893	0,928	0,788	0,858	0,619	0,407	0,425	0,416
1000	38	TRUE	10	0,903	0,930	0,882	0,906	0,618	0,406	0,424	0,415
1250	3	FALSE	∞	0,725	0,756	0,564	0,660	0,596	0,487	0,408	0,448
1250	3	TRUE	∞	0,714	0,736	0,703	0,719	0,595	0,496	0,409	0,453
1250	1	FALSE	∞	0,693	0,697	0,527	0,612	0,584	0,464	0,392	0,428
1250	1	TRUE	∞	0,655	0,681	0,646	0,663	0,546	0,592	0,381	0,487
1250	3	FALSE	1	0,572	0,509	0,389	0,449	0,553	0,484	0,368	0,426
1250	3	TRUE	1	0,557	0,504	0,561	0,533	0,555	0,480	0,369	0,425
1250	1	FALSE	1	0,444	0,842	0,357	0,600	0,444	0,839	0,356	0,598
1250	1	TRUE	1	0,546	0,823	0,529	0,676	0,457	0,802	0,358	0,580
1250	3	FALSE	2	0,606	0,472	0,416	0,444	0,579	0,414	0,377	0,395
1250	3	TRUE	2	0,572	0,539	0,575	0,557	0,558	0,481	0,372	0,426
1250	1	FALSE	2	0,586	0,466	0,395	0,431	0,566	0,456	0,374	0,415
1250	1	TRUE	2	0,559	0,679	0,546	0,613	0,505	0,613	0,357	0,485
1250	3	FALSE	3	0,575	0,667	0,413	0,540	0,541	0,581	0,376	0,479
1250	3	TRUE	3	0,598	0,722	0,577	0,649	0,523	0,638	0,372	0,505
1250	1	FALSE	3	0,502	0,785	0,379	0,582	0,482	0,782	0,368	0,575
1250	1	TRUE	3	0,576	0,783	0,552	0,668	0,456	0,824	0,360	0,592
1250	3	FALSE	5	0,649	0,634	0,478	0,556	0,587	0,490	0,400	0,445
1250	3	TRUE	5	0,649	0,697	0,635	0,666	0,570	0,545	0,393	0,469
1250	1	FALSE	5	0,595	0,586	0,421	0,503	0,576	0,494	0,390	0,442
1250	1	TRUE	5	0,593	0,719	0,573	0,646	0,493	0,717	0,365	0,541
1250	3	FALSE	10	0,707	0,753	0,541	0,647	0,589	0,504	0,404	0,454
1250	3	TRUE	10	0,703	0,746	0,685	0,716	0,586	0,517	0,403	0,460
1250	1	FALSE	10	0,663	0,692	0,494	0,593	0,586	0,441	0,391	0,416
1250	1	TRUE	10	0,639	0,667	0,630	0,648	0,549	0,576	0,381	0,479
1500	3	FALSE	∞	0,719	0,772	0,554	0,663	0,591	0,494	0,402	0,448
1500	3	TRUE	∞	0,716	0,741	0,703	0,722	0,593	0,493	0,405	0,449
1500	1	FALSE	∞	0,670	0,705	0,500	0,603	0,570	0,545	0,392	0,468
1500	1	TRUE	∞	0,660	0,697	0,646	0,671	0,574	0,510	0,389	0,449
1500	3	FALSE	1	0,581	0,481	0,390	0,435	0,561	0,470	0,370	0,420
1500	3	TRUE	1	0,557	0,537	0,555	0,546	0,542	0,521	0,364	0,442
1500	1	FALSE	1	0,444	0,844	0,356	0,600	0,443	0,840	0,355	0,598
1500	1	TRUE	1	0,544	0,845	0,525	0,685	0,443	0,840	0,355	0,598
1500	3	FALSE	2	0,614	0,443	0,419	0,431	0,587	0,400	0,380	0,390
1500	3	TRUE	2	0,572	0,572	0,568	0,570	0,545	0,527	0,368	0,447
1500	1	FALSE	2	0,587	0,466	0,393	0,430	0,566	0,456	0,372	0,414
1500	1	TRUE	2	0,556	0,484	0,561	0,523	0,554	0,491	0,368	0,429
1500	3	FALSE	3	0,583	0,637	0,415	0,526	0,549	0,556	0,377	0,466
1500	3	TRUE	3	0,600	0,689	0,582	0,636	0,535	0,610	0,375	0,492
1500	1	FALSE	3	0,502	0,786	0,378	0,582	0,482	0,782	0,367	0,575
1500	1	TRUE	3	0,571	0,796	0,546	0,671	0,480	0,775	0,365	0,570
1500	3	FALSE	5	0,639	0,682	0,468	0,575	0,574	0,539	0,394	0,466
1500	3	TRUE	5	0,652	0,698	0,636	0,667	0,575	0,550	0,397	0,473

Tableau A5 (suite) - Performance des arbres selon les jeux de paramètres

Paramètres				Scores - Entraînement				Scores - Test			
nbSols	hclust	Smote	limTree	Tx réussite	Rappel	Précision	$\bar{F}$	Tx réussite	Rappel	Précision	$\bar{F}$
1500	1	FALSE	5	0,569	0,675	0,408	0,542	0,544	0,639	0,386	0,512
1500	1	TRUE	5	0,593	0,708	0,573	0,641	0,529	0,656	0,378	0,517
1500	3	FALSE	10	0,707	0,766	0,540	0,653	0,585	0,508	0,399	0,454
1500	3	TRUE	10	0,709	0,749	0,691	0,720	0,585	0,510	0,399	0,455
1500	1	FALSE	10	0,650	0,710	0,479	0,595	0,555	0,604	0,388	0,496
1500	1	TRUE	10	0,649	0,661	0,642	0,652	0,577	0,492	0,389	0,441
1750	3	FALSE	∞	0,724	0,751	0,559	0,655	0,599	0,474	0,405	0,440
1750	3	TRUE	∞	0,718	0,730	0,708	0,719	0,598	0,487	0,406	0,447
1750	1	FALSE	∞	0,682	0,714	0,512	0,613	0,590	0,465	0,395	0,430
1750	1	TRUE	∞	0,671	0,696	0,658	0,677	0,592	0,472	0,397	0,435
1750	3	FALSE	1	0,578	0,489	0,387	0,438	0,559	0,458	0,364	0,411
1750	3	TRUE	1	0,556	0,560	0,550	0,555	0,535	0,530	0,359	0,445
1750	1	FALSE	1	0,433	0,865	0,352	0,609	0,433	0,859	0,351	0,605
1750	1	TRUE	1	0,543	0,845	0,523	0,684	0,442	0,840	0,353	0,597
1750	3	FALSE	2	0,590	0,508	0,401	0,454	0,561	0,452	0,365	0,409
1750	3	TRUE	2	0,572	0,524	0,573	0,548	0,563	0,454	0,366	0,410
1750	1	FALSE	2	0,546	0,583	0,376	0,479	0,494	0,710	0,362	0,536
1750	1	TRUE	2	0,557	0,564	0,551	0,557	0,502	0,694	0,364	0,529
1750	3	FALSE	3	0,583	0,641	0,413	0,527	0,550	0,548	0,374	0,461
1750	3	TRUE	3	0,600	0,693	0,580	0,636	0,536	0,596	0,372	0,484
1750	1	FALSE	3	0,502	0,763	0,373	0,568	0,475	0,772	0,360	0,566
1750	1	TRUE	3	0,568	0,745	0,546	0,645	0,483	0,756	0,362	0,559
1750	3	FALSE	5	0,643	0,666	0,469	0,567	0,576	0,507	0,388	0,447
1750	3	TRUE	5	0,648	0,691	0,631	0,661	0,575	0,537	0,392	0,465
1750	1	FALSE	5	0,552	0,682	0,394	0,538	0,512	0,657	0,365	0,511
1750	1	TRUE	5	0,591	0,694	0,571	0,632	0,522	0,634	0,368	0,501
1750	3	FALSE	10	0,713	0,753	0,545	0,649	0,591	0,493	0,400	0,447
1750	3	TRUE	10	0,710	0,731	0,697	0,714	0,592	0,503	0,403	0,453
1750	1	FALSE	10	0,646	0,706	0,473	0,589	0,586	0,494	0,396	0,445
1750	1	TRUE	10	0,657	0,644	0,655	0,650	0,594	0,458	0,397	0,428
2000	3	FALSE	∞	0,729	0,772	0,561	0,666	0,598	0,490	0,404	0,447
2000	3	TRUE	∞	0,709	0,721	0,698	0,709	0,598	0,477	0,402	0,439
2000	1	FALSE	∞	0,670	0,739	0,496	0,617	0,566	0,502	0,376	0,439
2000	1	TRUE	∞	0,659	0,681	0,645	0,663	0,569	0,506	0,380	0,443
2000	3	FALSE	1	0,550	0,572	0,376	0,474	0,531	0,523	0,353	0,438
2000	3	TRUE	1	0,555	0,567	0,545	0,556	0,533	0,528	0,355	0,442
2000	1	FALSE	1	0,432	0,865	0,350	0,608	0,431	0,859	0,349	0,604
2000	1	TRUE	1	0,539	0,863	0,519	0,691	0,433	0,856	0,349	0,603
2000	3	FALSE	2	0,589	0,511	0,399	0,455	0,564	0,454	0,365	0,410
2000	3	TRUE	2	0,571	0,519	0,569	0,544	0,563	0,451	0,363	0,407
2000	1	FALSE	2	0,559	0,549	0,378	0,464	0,499	0,700	0,362	0,531
2000	1	TRUE	2	0,557	0,547	0,550	0,549	0,501	0,698	0,362	0,530
2000	3	FALSE	3	0,593	0,572	0,410	0,491	0,567	0,493	0,375	0,434
2000	3	TRUE	3	0,596	0,709	0,572	0,641	0,531	0,616	0,369	0,492
2000	1	FALSE	3	0,554	0,612	0,384	0,498	0,532	0,556	0,359	0,458
2000	1	TRUE	3	0,568	0,729	0,545	0,637	0,481	0,760	0,360	0,560
2000	3	FALSE	5	0,644	0,633	0,466	0,549	0,586	0,497	0,393	0,445
2000	3	TRUE	5	0,648	0,666	0,635	0,651	0,582	0,522	0,393	0,457
2000	1	FALSE	5	0,555	0,672	0,393	0,532	0,520	0,636	0,364	0,500
2000	1	TRUE	5	0,586	0,690	0,565	0,627	0,523	0,630	0,366	0,498
2000	3	FALSE	10	0,712	0,764	0,541	0,652	0,592	0,511	0,401	0,456
2000	3	TRUE	10	0,706	0,728	0,690	0,709	0,594	0,492	0,400	0,446
2000	1	FALSE	10	0,623	0,753	0,453	0,603	0,530	0,617	0,368	0,492
2000	1	TRUE	10	0,637	0,710	0,613	0,661	0,541	0,601	0,373	0,487

DOCTORAT BRETAGNE LOIRE / MATHSTIC

Université Bretagne Sud

Titre : Résolution de problèmes réalistes de tournées à flotte hétérogène en milieu urbain : vers un solveur adaptatif mêlant recherche opérationnelle et apprentissage automatique.

Mot clés : Optimisation, analyse statistique, apprentissage automatique, métaheuristique, tournées de véhicules

Résumé : Dans le domaine de la recherche opérationnelle, la résolution des problèmes de tournées constitue un défi emblématique depuis les années 1960. Initialement développées sur des problèmes simples, de nombreuses variantes ont permis d'améliorer le réalisme des instances étudiées, en particulier en milieu urbain où la congestion dépend du véhicule utilisé. Cependant, encore peu de méthodes sont compatibles avec l'ensemble des variantes existantes, celles-ci perdant de l'efficacité avec l'augmentation du nombre de clients.

C'est dans ce contexte que nous avons développé un solveur robuste et adaptatif, obtenant de bonnes solutions dans un très bref délai sur un nombre élevé d'instances aussi difficiles que variées (plusieurs milliers de clients, véhicules hétérogènes, milieu urbain, etc).

Ce nouveau solveur permet de générer des solutions issues de multiples instances, caractérisées par un ensemble de variables, dont l'étude permet de mieux comprendre ce que constitue une bonne solution.

Dans un contexte où les travaux interdisciplinaires croisant recherche opérationnelle et apprentissage automatique sont de plus en plus fréquents, nous nous sommes intéressés à l'utilisation d'arbres de décision pour cartographier les aires contenant a priori l'ensemble des solutions de bonne qualité. La combinaison entre les arbres de décision et ce nouveau solveur restreint la recherche de bonnes solutions aux zones prometteuses, afin d'augmenter les chances d'obtenir de bons résultats et de réduire la durée de cette exploration.

Les résultats expérimentaux indiquent que ces travaux interdisciplinaires sont prometteurs pour générer un ensemble de méthodes hybrides performantes.

Title: Solving realistic vehicle routing problems with heterogeneous fleet in urban environments: towards an adaptive solver combining operations research and machine learning.

Keywords: Optimisation, statistical analysis, machine learning, metaheuristic, vehicle routing

Abstract: In the field of operations research, solving vehicle routing problem has been an emblematic challenge since the 1960s. Initially developed on simple problems, many variants have improved the realism of the studied instances, especially in urban areas where congestion depends on the vehicle used. However, there are still few methods consistent with all the existing variants, which are becoming less effective as the number of clients increases.

It is in this context that we have developed a robust and adaptive solver, obtaining good solutions in a very short period of time on a high number of instances as difficult as varied (several thousand customers, heterogeneous vehicles, urban environment, etc).

This new solver makes it possible to generate solutions from multiple instances, characterized by a set of variables, which allows a better understanding of what constitutes a good solution.

In a context where interdisciplinary work combining operational research and machine learning is becoming more and more frequent, we were interested in the use of decision trees to map areas containing a priori all good quality solutions. The combination between decision trees and this new solver restricts the search for good solutions to promising areas, in order to increase the probability of obtaining good results and to reduce the duration of this exploration.

Experimental results indicate that this interdisciplinary work is promising to generate a set of efficient hybrid methods.